ARO-D REPORT 71-4

PROCEEDINGS OF THE 1971 ARMY NUMERICAL ANALYSIS CONFERENCE

SPONSORED BY
THE ARMY MATHMATICS STEERING COMMITTEE ON BEHALF OF

THE OFFICE OF
THE CHIEF OF RESEARCH AND DEVELOPMENT

U. S. Army Research Office-Durham

Report 71-4

December 1971

PROCEEDINGS OF THE 1971 ARMY NUMERICAL ANALYSIS CONFERENCE

Sponsored by the Army Mathematics Steering Committee

Host

Department of Defense Computer Institute

Washington Navy Yard

Washington, D. C.

U. S. Army Research Office-Durham
Box CM, Duke Station
Durham, North Carolina

FOREWORD

In a conversation with Roger MacGowan, Dr. John H. Giese, Chairman of the Numerical Analysis Conference, raised the question of holding the 1971 Army Numerical Analysis Conference at the Department of Defense Computer Institute, Washington Navy Yard, Washington, D. C. Mr. MacGowan, formerly with the Army Missile Command at Redstone Arsenal, is now with DODCI. He has been an active participant in this series of conferences and was willing to discuss the possibility of his installation serving as host for the 1971 Army Numerical Analysis Conference with CPT George P. Sotos USN, Director of DODCI. The facilities at the Washington Navy Yard were exceptionally fine for holding a conference of this size, and those in attendance are indebted to CPT Sotos for agreeing to host it at his installation. They are also indebted to Roger MacGowan for serving as Chairman on Local Arrangements. He and LTC F. D. Troyan did an outstanding job in handling the many problems that arose during the conduction of the conference.

The following information taken from a folder issued by DODCI will serve to acquaint interested individuals in the services offered by the host of this conference. "The Department of Defense Computer Institute is a jointly staffed activity established by the Secretary of Defense under the Executive Agency of the Secretary of the Navy. The Institute functions under the sponsorship of the Chief of Naval Operations. The Department of Defense Computer Institute provides computer orientation courses which are designed to acquaint SENIOR MILITARY AND CIVILIAN DOD EXECUTIVES with the application, operation and selection of digital computer systems. The courses provide a comprehensive view of the computer field, and are directed to:

1. Teaching the fundamentals of digital computer capabilities, applications, and limitation.

2. Planning and implementation of new digital computer systems and improving existing systems.

3. Enabling DOD to plan and operate its systems more independently of contractors.

Requests for information should be addressed to:

Director
Department of Defense Computer Institute
Bldg, 175
Washington Navy Yard
Washington, D. C. 20390."

The theme of the invited addresses was Computer Aided Design and Engineering. Dr. James H. Griesmer with International Business Machines Corporation was the lead-off speaker. His address was entitled "Scratchpad/1: An Interactive Facility for Symbolic Mathematics." Professor J. C. R. Licklider, Director of Project MAC, spoke on "Modern Facilities for Man Computer Interaction." Professor A. Van Dam, Brown University, concluded the conference with a talk on "Low Cost Interactive Computer Graphics." Besides these invited addresses, there were fourteen informative contributed papers.

The Army Mathematics Steering Committee sponsors these conferences on behalf of the Office of the Chief of Research and Development. Members of this committee have asked that most of the papers presented at this symposium be printed in these Proceedings in order that persons unable to attend the meeting may become acquainted with their informative contents. They would like to thank Dr. John H. Giese and members of his Program Committee for organizing this 1971 Army Numerical Analysis Conference. They appreciate very much the time and efforts of the many speakers and chairmen who really made this conference such an interesting and scientific event.

TABLE OF CONTENTS

**These papers were presented at the conference but do not appear in the Proceedings.

A G E N D A

1971 ARMY NUMERICAL ANALYSIS CONFERENCE*
Department of Defense Computer Institute
Washington Navy Yard, Washington, D. C.

Thursday, 22 April 1971

0800-0900 REGISTRATION - Building 175

0900-0920 OPENING OF THE CONFERENCE

Roger MacGowan, Local Representative, Department of Defense Computer Institute, Washington Navy Yard, Washington, D. C.

WELCOMING REMARKS

Captain George P. Sotos, USN, Director of the Department of Defense Computer Institute

0920-1020 GENERAL SESSION I

Chairman: Roger MacGowan, Department of Defense Computer Institute

SCRATCHPAD/1: AN INTERACTIVE FACILITY FOR SYMBOLIC MATHEMATICS

Dr. James H. Griesmer, International Business Machines Corporation, Yorktown Heights, New York and the University of California at Berkeley, Berkeley, California

1020-1050 BREAK

1050-1220 TECHNICAL SESSION 1**

Chairman: Sylvan Eisman, Frankford Arsenal, Philadelphia, Pa.

EXPEDITING LARGE SCALE NUMERICAL COMPUTATIONS BY THE USE OF TV TECHNIQUES

Jenny Bramley, U. S. Army Engineer Topographic Laboratories, Fort Belvoir, Virginia

COMPUTER SIMULATION: AN ESSENTIAL APPROACH TO SOLVE COMPLEX BIOLOGICAL AND ENVIRONMENTAL PROBLEMS

F. Heinmets, Biophysics Group, Pioneering Research Laboratory, U. S. Army Natick Laboratories, Natick, Massachusetts

---

*The following local representatives will be glad to assist you with various problems: Mr. Roger MacGowan, LTC Dan Troyan, USA.

**Technical Sessions 1 and 2 will run concurrently.

1050-1220 TECHNICAL SESSION 2

Chairman: Captain Clay Smith, Mathematics Branch, Office of the Chief of Research and Development

MATHEMATICAL PROGRAMMING MODELS OF AN OPTIMAL DELAYING ACTION

CPT John A. Battilega and LT William Moskowitz, U. S. Army Strategy and Tactics Analysis Group, Bethesda, Maryland

THE USE OF MULTIPLE REGRESSION EQUATIONS IN THE CAMPAIGN EXECUTION MODEL

Sol Haberman, The Johns Hopkins University, Applied Physics Laboratory, Silver Spring, Maryland

1220-1330 LUNCH

1330-1520 TECHNICAL SESSION 3(Sessions 3 and 4 will run concurrently.)

Chairman: B. F. Caviness, Mathematics Research Center, University of Wisconsin, Madison, Wisconsin

DESIGN OF SURVEY SYSTEMS USING NONLINEAR PROGRAMMING METHODS

CPT Clifford W. Greve, U. S. Army Engineer Topographic Laboratories, Fort Belvoir, Virginia

BOUNDS FOR ERROR IN THE SOLUTION SET OF A PERTURBED LINEAR PROGRAM

Stephen M. Robinson, Mathematics Research Center, University of Wisconsin, Madison, Wisconsin

A MODIFIED ALLOCATION METHOD FOR COMPUTING STRESS INTENSITY FACTORS

M. A. Hussain and W. E. Lorensen, Maggs Research Center, Watervliet, New York

1330-1520 TECHNICAL SESSION 4(Sessions 3 and 4 will run concurrently.)

Chairman: H. S. Hung, Mathematics Research Center, University of Wisconsin, Madison, Wisconsin

FINITE ELEMENT ANALYSIS OF BURIED CYLINDERS

J. L. Kirkland, U. S. Army Engineer Waterways Experiment Station, Vicksburg, Mississippi

APPLICATION OF SPLINE INTERPOLATING METHODS TO ENGINEERING PROBLEMS

James B. Cheek, Jr., Narayanaswamy Radhakrishnan, and Fred T. Tracy, U. S. Army Waterways Experiment Station, Vicksburg, Mississippi

A SPLINE APPROXIMATION OF AN EMPIRICAL DISPLACEMENT FIELD

R. D. Scanlon, Maggs Research Center, Watervliet, New York

1520-1550 BREAK

1550-1650 GENERAL SESSION II

Chairman: Dr. Siegfried Lehnigk, Physical Sciences Laboratory, U. S. Army Missile Command, Redstone Arsenal, Alabama

MODERN FACILITIES FOR MAN COMPUTER INTERACTION

Professor J. C. R. Licklider, Director, Project MAC, Cambridge, Massachusetts

Friday, 23 April 1971

0830-1050 TECHNICAL SESSION 5

Chairman: Jagdish Chandra, Mathematics Division, Army Research Office-Durham, Durham, North Carolina

A CORRECTING PROCESS DESIGNED TO CONTROL PROPAGATED ERROR

CPT Isaac S. Metts, Jr., Walter Reed Army Institute of Research, Walter Reed Army Medical Center, Washington, D. C.

ITERATIVE SOLUTION OF ABSTRACT POLYNOMIAL EQUATIONS

L. B. Rall, Mathematics Research Center, University of Wisconsin, Madison, Wisconsin

ROUNDING

J. M. Yohe, Mathematics Research Center, University of Wisconsin, Madison, Wisconsin

NUMERICAL SPECTRA AND APPLICATIONS TO COMPUTATIONAL METHODS

M. Z. Nashed, Mathematics Research Center, University of Wisconsin, Madison, Wisconsin

1050-1120 BREAK

1120-1220 GENERAL SESSION III

Chairman: Joseph Weinstein, Systems Cost Analysis Office, U. S. Army Electronics Command, Fort Monmouth, New Jersey

LOW COST INTERACTIVE COMPUTER GRAPHICS

Professor A. Van Dam, Brown University, Providence, Rhode Island

SCRATCHPAD/1 AN INTERACTIVE FACILITY FOR SYMBOLIC MATHEMATICS

James H. Griesmer*
IBM Thomas J. Watson Research Center
Yorktown Heights, New York

and

University of California
Berkeley, California

1. INTRODUCTION. In the past few years considerable interest has been shown in the use of computers to carry out symbolic mathematical computations. A recent conference, the Second Symposium on Symbolic and Algebraic Manipulation < 9 >, contained nearly 50 papers on the subject, including several papers on computer systems which provide facilities for algebraic computation. SCRATCHPAD/1 is one such system, distinguished both by its powerful symbolic capability, and by its user language containing notations which resemble those of conventional mathematics.

SCRATCHPAD/1 has been implemented in the LISP programming language using an experimental System/360 LISP system. The principal features of this LISP system which enhance its capability for symbolic and algebraic computations are provisions for unlimited precision integer arithmetic and for accessing a sizable number of LISP programs.

This latter capability has enabled other symbolic systems written in LISP to be incorporated into the SCRATCHPAD library. Significant portions of the following systems are simultaneously available to the SCRATCHPAD user: REDUCE2 < 4 >, MATHLAB < 2 >, SIN < 8 >, and Korsvold's On-Line Simplification System < 6 >.

An interactive version of SCRATCHPAD is available to users on a S/360 Model 67 under the CP/CMS time-sharing system; a batch version has been implemented on a S/360 Model 91 under OS/360.

2. SYSTEM ORGANIZATION AND CAPABILITIES. SCRATCHPAD may be viewed as having four components:

an input translator to convert the input strings (commands) from the user into a form suitable for interpretation;

an output translator to convert expressions from an internal form to a two-dimensional format for output to the user;

---

*Visiting Mackay Lecturer, Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, 1970-71.

a library containing the bulk of the algebraic manipulation facilities; and

an evaluator which causes the commands issued by the user to be carried out.

(i) The input translator. User commands in the source language of SCRATCHPAD are accepted by the input translator and converted into a form suitable for interpretation by the evaluator. The current mode of input for interactive use of the system is via an IBM 2741 communications terminal.

The SCRATCHPAD language allows mathematical entities with two-dimensional graphics; e.g.,

$$x_i(t) \qquad y^{i}_{j,k} \qquad \sum_{j=1}^{n} j^2$$

For keyboard input all two-dimensional forms are linearized in a straight-forward manner; for example, the above entities are linearized as follows:

```
x< i >(t)          y<j,k;i;;1>          sum <j=1;n>j**2
```

(ii) The output translator. A modification of the CHARYBDIS program from MATHLAB < 7 >, is used in SCRATCHPAD to produce a two-dimensional image of a mathematical expression. CHARYBDIS output may be obtained either from a 2741 terminal or from any other line printing output device.

(iii) The library of SCRATCHPAD. The following symbolic capabilities, most of which were originally written for other systems, are currently available in SCRATCHPAD/1:

Simplication (R,M,K,S,N)
Polynomial Greatest Common Divisor (R)
Differentiation (R,N)
Integration (M,S)
Polynomial Factorization (M)
Direct and Inverse Laplace Transforms (M,K)
Matrix Operations (R,N)
Solutions of Systems of Linear Equations (N)

The letter "R,M,K,S" refer respectively, to REDUCE2, MATHLAB, Korsvold, and SIN. The letter "N" refers to newly created facilites.

The following is indicative of the variety of problems for which the library has been utilized:

(a) generation and study of polynomials arising in graph theory and sorting;

(b) inversion of transition matrices resulting from a problem in data compression;

(c) symbolic differentiation and substitution required in a study of wave propagation in an elastic media;

(d) symbolic triple integrations arising in queuing theory;

(e) solution of an eight dimensional system of linear symbolic equations arising from research on optimal difference formulae;

(f) investigation of subdeterminants derived from transformations applied to systems of nonlinear ordinary differential equations.

(iv) The SCRATCHPAD evaluator. Evaluation consists of a systematic transformation of an entity such as a mathematical expression, as governed by the current "environment", that is, the set of all definitions, rules, and flag settings in effect at a particular instant in time. The SCRATCHPAD system provides the user with considerable control over the evaluation process, by allowing him to introduce new substitution rules and pattern-matching rules, and to modify system flags and variables.

3. THE SCRATCHPAD LANGUAGE. The SCRATCHPAD language is designed primarily for interactive use by a mathematician unskilled as a programmer. It features a syntax which is simple and concise, yet rich in mathematical constructs. Its design currently provides a framework for manipulation of all of the following:

(a) finite and infinite sums, products, and sequences;

(b) relations such as equations and inequalities;

(c) arbitrarily indexed variables, functions, and operators;

(d) sets;

(e) arrays of arbitrary dimension.

In this paper we shall mention some of the essential features of the SCRATCHPAD language. A more complete description of the user language is contained in < 3 >.

In order to illustrate some of the features of the SCRATCHPAD language, we shall consider the calculation of Legendre polynomials using the recurrence relation:

$$p_n = \frac{2n-1}{n} \cdot x \cdot p_{n-1} - \frac{n-1}{n} \cdot p_{n-2}$$

subject to the initial conditions

$$p_0 = 1, \quad p_1 = x.$$

To enter this recursive definition into the system, the SCRATCHPAD user need merely type the following four STATEMENTS, issued as individual COMMANDS, using the above rules for linearization:

```
p<0> = 1

p<1> = x

n in (2,3,...)

p<n> = ((2*n-1) * x * p<n-1> - (n-1) * p<n-2>)/n
```

To get the first 5 Legendre polynomials as output, the user types the COMMAND:

```
p<n> , n in (0,1,...,4)
```

and the system responds with

$$p_0 : 1$$

$$p_1 : X$$

$$p_2 : \frac{3X^2 - 1}{2}$$

and so on.

The above example illustrates several aspects of the SCRATCHPAD language besides its closeness to conventional mathematical notation. STATEMENTS are the fundamental constructs for making definitions and declarations. They consist of:

(i) a left part: usually a VARIABLE or a FORM (e.g., x < i >, or f(x));

(ii) a relator: >, >=,=, <=, <, in;

(iii) a right part: an EXPRESSION.

VARIABLES and FORMS, together with (unlimited precision) INTEGERs, VECTORs, and SETs make up the PRIMITIVEs of the language. PRIMITIVEs combine with infix and prefix operators to form EXPRESSIONs.

The ellipsis (...) may be used in sums, products, and sequences to indicate missing terms; e.g.,

```
1 + 2 + ... + n              j in (1,2,...)
```

STATEMENTs may be issued single as COMMANDs:

```
j in (1,2,...)
```

in which case they are in effect during the evaluation of all subsequent COMMANDs. On the other hand, several STATEMENTs may be issued as a single COMMAND:

```
f(x) = x**j,  x > 0, j in (1,2,...)
```

Here the STATEMENTs on the right affect only the environment for evaluation of STATEMENTs to their left. As suggested by the examples above, STATEMENTs are used to assign a value, or more generally a range of possible values to a VARIABLE or FORM.

The SCRATCHPAD notations for VECTORs has been derived from SYMBAL < 1 >. The elements of a VECTOR can be labeled:

```
(1: 1 + x, 2:  1 + 2*x, 3:  1 + 3*x, 4:  1 + 4*x)
```

or unlabeled:

```
(1 + x, 1 + 2*x, 1 + 2*x, 1 + 4*x)
```

Ellipses can be used in a variety of ways; e.g., to indicate an infinite VECTOR or sequence:

```
(1,3,5,...)
```

Matrices are viewed as VECTORs of (row) VECTORs:

a 2 x 2 matrix:	((a,b), (c,d))
the n-dimensional identity matrix	(i: (j: i = j)), (i,j) in (1,2,...,n)

4. USER CONTROL. After a COMMAND is typed into the system, its constituent STATEMENTs are evaluated from right to left. Evaluation of EXPRESSIONSs in STATEMENTs consists of several phases, including simplification, substitution, expansion, pattern matching, and expression restructuring.

SCRATCHPAD provides the user with a wide variety of controls over the evaluation process in the form of flags, special variables, and selection capabilities. For example, by controlling the value of the special variable EXP, he may control whether EXPRESSIONs such as

```
(x+5) * (x-3)      or      (x+4) ** 10
```

will be expanded in full.

Other special variables allow the user to control the formatting of expressions; e.g., the STATEMENT

```
ORDER  = (z,y)
```

indicates that 'z' is to occur before 'y' in the output of polynomial products. Also the variable FACTORS can be set by the user to a list of those variables which are to be factored out of expressions on output.

If the user wishes to have previous results saved on a "workfile" on secondary storage, he can set the value of the special variable HISTORY to 1. Any previously computed result can then be recalled by referring to its integer label.

Another aspect of the user control is the availability of the WHERE-clause to qualify an EXPRESSION. For example,

```
... + (q(x) where x=b) + ...
```

causes q(x) to be evaluated at the point x=b, regardless of any previous value that x was assigned.

Other facilities enable the user to access sub-expressions or coefficients of a variable within an EXPRESSION, e.g.,

```
coeff (x,3,x**4-7*x**3+5)
```

evaluates to -7, the coefficient of x raised to the third power.

5. EXTENSION FACILITIES. A number of ways of extending SCRATCHPAD are available. Of these we briefly mention here the facilities provided for user-defined procedures and syntax extension.

To give the user the capability of executing a block of COMMANDs over and over, a facility for creating procedures is provided. Any COMMAND, preceded by a label of the form "n,m" for n and m integers, e.g.,

```
1.40      s=s+b*(y where x = a)
```

is neither translated nor interpreted, but rather stored by the system for later reference. The integer "n" is the procedure number, and "m" the COMMAND number. The user may type the constituent COMMANDs into SCRATCHPAD in any order. When all COMMANDs have been entered the user issues a COMMAND to create the procedure for subsequent interpretation.

As an example, using the above recurrent relation for defining Legendre polynomials, a procedure for obtaining the Legendre polynomial of degree n, p < n >, can be written as follows:

```
1.10          p< 0 > = 1
1.20          p< 1 > = x
1.30          m = 2
1.40          return p< n >  if m > n
1.50          p< m > = ((2*m-1) * x * p< m-1 >
                - (m-1) * p < m-2 > )/m
1.60          m=m+1
1.70          go  40
```

together with

```
legendre (n) = procedure (1), m local
```

Then the COMMAND

```
legendre (6)
```

will cause p< 6 > to be calculated and typed out.

COMMAND steps in procedures may be inserted, replaced or deleted. In the above procedure, if the user also wished to have p < m >, m < = n, typed out, he need merely add the COMMAND

1.55 p< m >

followed again by:

legendre (n) = procedure (l), m local

The added COMMAND would be inserted between COMMANDs 50 and 60.

An important objective of any system which hopes to answer the needs of mathematicians working in diverse areas is to enable a user to introduce and use the specialized notations from his problem area. To accomplish this objective, a syntax extension facility has been provided in SCRATCHPAD. This facility enables a user to extend the base language in a convenient way. For example, the notation in the base language for the absolute value of an EXPRESSION x is 'absval (x)'. If the user wishes to use the notation |x| for absolute value, then he may issue the COMMAND:

"|x|" = "absval (x)", x expression

The general form of a syntax extension COMMAND is

"N" = "D", < qualifier >, < qualifier >, ...

where N is a new notation, and D is its definition in terms of known constructs in the base language plus all extensions to date. Further details on the syntax extension feature are contained in < 5 >.

6. ACKNOWLEDGEMENT. Members of the SCRATCHPAD project at the T. J. Watson Research Center are Fred W. Blair, Richard D. Jenks, and the author. During the past year, while the author has been on leave at the university of California at Berkeley, SCRATCHPAD/1 was brought to its present level of development through the efforts of the author's two colleagues.

F. W. Blair is responsible for the design of the experimental S/360 LISP system. The design and implementation of the language, evaluator, and syntax extension feature is due to R. D. Jenks. The collection and integration of the numberous subsystems into the SCRATCHPAD library and the implementation of the unlimited precision integer arithmetic package is the work of the author. Modifications to CHARYBDIS were made by F. W. Blair.

The dependence of SCRATCHPAD on the work of Anthony C. Hearn (REDUC2), Carl Engelman (MATHLAB), Joel Moses (SIN), Jonathan K. Millen (CHARYBDIS), and Knut Korsvold (On-Line Simplification System) has already been mentioned. These individuals were most helpful in making their systems available in providing assistance which enabled us to utilized their work more easily.

REFERENCES

<1 > Engeli, M.E., "User Manual for the Formula Manipulation Language SYMBAL", Computation Center, University of Texas, March 1968.

<2 > Engelman, C., "The Legacy of MATHLAB 68", in < 9 >.

<3 > Griesmer, J. H., and Jenks, R. D., "SCRATCHPAD/1 - An Interactive Facility for Symbolic Mathematics", in < 9 >. (Also available as IBM Research Report RC 3260.)

<4 > Hearn, A. C., "REDUCE2: A System and Language for Algebraic Manipulation", in < 9 >.

<5 > Jenks, R. D., "META/PLUS - THe Syntax Extension Facility for SCRATCHPAD", to appear in the Proceedings of the 5th International Congress of the International Federation on Information Processing Ljubljana, Yugoslavia, August 23-28, 1971. (Also available as IBM Research Report RC 3259.)

<6 > Korsvold, L., "An On-Line Algebraic Simplify Program", Stanford Artificial Intelligence Project Memorandum No. 37, Stanford University, November 1965.

<7 > Millen, J. K., CHARYBDIS: A LISP Program to Display Mathematical Expressions on Typewriter-like Devices", in Interactive Systems for Experimental Applied Mathematics, M. Klerer and J. Reinfelds, eds., Academic Press, New York 1968, pp. 79-90.

<8 > Moses, J., "Symbolic Integration", Project MAC Report MAC-TR-47 (Thesis), Massachusetts Institute of Technology, Cambridge, Massachusetts, December 1967.

<9 > Petrick, S. R., ed., Proceedings of the Second Symposium on Symbolic and Algebraic Manipulation, Association for Computing Machinery, New York, March 23-25 1971.

EXPEDITING LARGE SCALE NUMERICAL COMPUTATIONS

BY THE USE OF TV TECHNIQUES

JENNY BRAMLEY
Computer Sciences Laboratory
US Army Engineer Topographic Laboratories
Fort Belvoir, Virginia 22060

This is a preliminary report limited to the presentation of the fundamentals involved in implementing the use of TV techniques for carrying out numerical computations. The completed work will be published in a periodical primarily devoted to computers. The method described is mainly intended for the correlation and processing of pictorial information. Thus it operates on experimental inputs having as a rule an accuracy of 1-2%.

The search for an alternative to the ubiquitous digital computer was motivated by the delays inherent in the input and output functions, which offset the high speed of digital computaion and result in high cost of operation. The delays are further compounded by the need for piecewise operation since storage of intensity data pertaining to even a relatively small picture, e.g., 10^6 picture elements quantized to 64 gray levels, cannot be accomplished cost-effectively in an inner computer memory.

The alternative selected is based on the use of a cathode ray tube (CRT) in conjunction with equipment for light flux measurement

to perform fully automated multi-dimensional mensuration. The method will be explained below on the evaluation of the correlation integral

$$I(s) = \int_{a}^{a+L} f(x)g(x-s)dx.$$

The analog system needed for this purpose includes the following components:

1. A TV monitor for about 1,000 line presentation with a CRT having a screen of very short persistence.

2. A detector, which integrates the light flux received during one frame period T, together with the associated optical system to focus the light output of the CRT on it.

3. Small scale video storage, such as a video disc.

4. Means for the utilization of the results obtained. These would either be displays for direct viewing, or the output could be digitized and fed into a digital computer. Still another possibility of special interest in correlation work is to use the output as a control for further processing.

The use of TV techniques imposes a number of constraints on the variables: All quantities must be single-valued, positive and normalized to fall within specified ranges. The first requirement is basic, the second one can be satisfied by adding a bias constant, which is then subtracted from the final result, while the third one merely calls for suitable units.

All the mensuration operations are performed within the CRT raster ABCD, a square of side L, as shown in Figure 1. The vari-

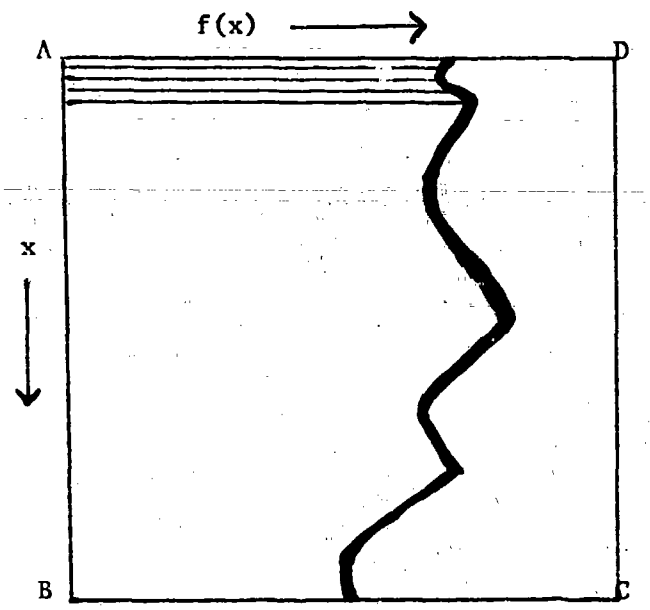


FIGURE 1

able x along the line AB can take on all integral values from 1 to N, the total number of raster lines. (In mathematical terms, a scan or raster "line" is a narrow band comprised between two straight lines.) A suitable value is N∿1000, readily available in commercial high resolution TV systems.

To plot a function f(x), adjust the scale so that the numerical value of f(x) falls between 0 and L. It is not restricted to integral multiples of L/N (corresponding to N resolution points).

The plotting is performed by scanning with the electron beam along all successive horizontal scan lines numbered 1,...,n,...,N. Along any line n, the beam current is cut off at a vertical location whose distance from AB is f(n). This functional representation is rotated by 90° from conventional mathematical plots to conform to the conventional horizontal scanning of a CRT.

Integration under the curve f(x) is performed by an integrating light detector which measures the light output from the CRT screen during a frame time $T \ll 1/30$ sec of conventional TV. The screen persistence must be shorter than the scan duration of one line, otherwise the lines that are scanned first contribute more to the integrated light output than those that are scanned last. Under these conditions, the light flux is proportional to $\sum_{n=1}^{N} f(n)L/N$, the conventional expression for the numerical evaluation of the integral $\int_{a}^{a+L} f(x)dx$.

This procedure is applicable to the evaluation of the integral $I(s) = \int_{a}^{a+L} f(x)g(x-s)dx$, i.e., of the sum $\sum_{n=1}^{N} f(n)g(n-s)L/N$ approximating it, provided g(x-s) represents the brightness of the line specified by x. (Obviously, for every value of s, there is a different set of g values.) Any particular term in the sum, e.g.,

f(n)g(n-s)L/N represents the light flux from scan line n, of width L/N, scanned at a brightness g(n-s) for a length f(n) starting from the left edge of the raster, i.e., from the line AE in Figure 1. Therefore, the light flux integrated over the frame time T represents a numerical value for the integral I(s). The function g originates as a video signal V applied to the circuitry of the CRT. To be able to generate on command any desired value of the line brightness, it is necessary that g be proportional to V. This is accomplished by passing the video signal through a gamma amplifier which matches the gamma of the CRT.

As far as the speed of operation is concerned, each of the N products in the sum representing I(s) can be readily formed in 1 μsec, particularly if the CRT is of the electrostatic deflection type. The 1 μsec time interval takes care of the writing rate of the tube, the response of the phosphor screen, and the response of the photomultiplier. The retrace time from the end of one scan line to the beginning of the next one is a fraction of 1 μsec. If f(x) is a processing function and is recorded on a video disc, there is no difficulty in sensing it during the retrace time, particularly since, by definition, f(x) is a single step function on every scan line. Therefore, the sum of 1000 terms will require little more than 1 millisecond. If the values of g are obtained by scanning the picture with a flying spot scanner, that operation can be slowed down to synchronize with the summation process.

If I(s) represents a correlation integral in the comparison of two sets of pictorial data, then f(x) is also an intensity and is introduced into the system as a grid voltage drive. Since the procedure requires that f(x) specify the length of a scan, the signal must be transformed to the scale of a horizontal deflection voltage by an operational amplifier. Under ordinary circumstances intensities are not obtainable to 0.1%, so that it is superfluous to use a line of 1000 resolution elements to represent the maximum intensity value. A much shorter scan length can be used with attendant increase in speed. Thus 1 μsec will cover both the writing and the retrace time, allowing 1000 correlations per second of 1000 points each.

Another mathematical operation of special interest in image processing is the Fourier transform. Let g(n,m) (n and m ranging from 1 to N) represent the intensities of the N^2 successive elements of a picture arranged in N rows and columns. It is a multiple step function, constant over the area of any element and changing only at its boundaries. The sine and cosine transforms of the function g(n,m) are given by

$$C(n,\mu) = \sum_{m=1}^{N} g(n,m) \cos 2\pi m\mu/N,$$

$$S(n,\mu) = \sum_{m=1}^{N} g(n,m) \sin 2\pi m\mu/N,$$

$$C(n,\mu) + iS(n,\mu) = G(n,\mu),$$

but these sums cannot be formed directly by the flux integration method since sines and cosines can assume both positive and negative values.

We introduce therefore, the processing functions

$$f_c(m,\mu) = 1 + \cos 2\pi m\mu/N,$$

$$f_s(m,\mu) = 1 + \sin 2\pi m\mu/N,$$

which give

$$C(n,\mu) + \sum_{m=1}^{N} g(n,m) = \sum_{m=1}^{N} g(n,m)F_c(m,\mu),$$

$$S(n,\mu) + \sum_{m=1}^{N} g(n,m) = \sum_{m=1}^{N} g(n,m)F_s(m,\mu).$$

The summation term on the left represents the sum of the intensities of all the picture elements on line n. It is a bias level added to the Fourier transform to keep it positive. The sums on the right are formally identical with the samerepresenting the integral I(s) and can be evaluated in exactly the smae way.

Presumably, the purpose of a Fourier transform is a filtering operation in frequency space. The procedure, therefore, is as follows:

1. Perform the Fourier transform of the input.
2. Write the results on the video disc as they are being obtained.

3. Read the results off the disc in order to multiply them by the filter function provided as one of the inputs.

4. Store this second set of results on the video disc.

5. Read the results off the disc and transform them back to ordinary coordinate space.

While there may be degradation each time information is written on the disc and is read back, it should not be significant with a good quality disc and associated electronics. However, additional transfers to and from storage may have to be limited as long as the electronic components are limited to those readily available.

Consider now the consequences of these limitations on the processing times involved. If every real or imaginary term in the summation takes 1 μsec, then each term in the corresponding transform is formed in 1 msec, so that about 17 minutes are needed for a one-dimensional transform of a $10^3 \times 10^3$ element picture, counting both input and output times. Fourier transforms on a digital computer are carried out using the Cooley-Tukey algorithm, this method constitutes, therefore, the standard of comparison for the time required to perform the operation. Multiplexing is possible on a CRT computer, though the auxiliary coefficients that have to be computed require a large number of transfers to and from storage. The parameters of the video disc and its associated equipment needed for the Cooley-Tukey approach with specified limits of

error will be discussed in a subsequent paper. However, the first step of this approach may be practical at present. If $N = r^2$, we break up the N term summation in the Fourier transform into two summations of r terms each. Thus, for every value of n and μ, there are 2r, i.e., $2\sqrt{N}$ operations to be performed.

$$m = m' + m''r, \qquad 0 \leq m'' < r-1, \qquad 1 \leq m' \leq r$$

$$\mu = \mu' + \mu''r, \qquad 0 \leq \mu'' < r-1, \qquad 1 \leq \mu' \leq r$$

$$G(n,\mu',\mu'') = \sum_{m'=1}^{r} W^{m'(\mu' + \mu''r)} \left[\sum_{m''=0}^{r-1} g(n,m' + m''r) W^{m''\mu' r} \right]$$

$$W = e^{2\pi i}$$

If $N = 1024$ or 32^2, the processing time for the entire picture is cut down from 1024^3 μsec to 64×1024^2 μsec, which is just in excess of one minute. The regular Cooley-Tukey approach requires about 10 sec.

On the basis of the considerations presented here, I believe we may conclude that it is time to stop tying down expensive digital computers with the processing of vast amounts of experimental data. Except where high precision is required, it is much more cost effective to use TV type equipment. At the very least, it can serve as a first approximation to find data that merit more precise consideration.

COMPUTER SIMULATION: AN ESSENTIAL APPROACH TO SOLVE COMPLEX BIOLOGICAL AND ENVIRONMENTAL PROBLEMS

F. Heinmets

Pioneering Research Laboratory, U. S. Army Natick Laboratories, Natick, Mass.

When viewed at realistic level, the common property of many biological and environmental systems is that they are composed of numerous sets of entities entailed in multiplicity of modes of interactions. Both systems are dynamic. Time dependance of these phenomena is a basic feature and has to be accounted for in modeling process. We propose to consider here some essential features of modeling for both systems and we will attempt to point out difficulties envisioned in this type of work. It is self-evident, without the need of further elaboration, that only via computer simulation will it be possible to unravel the operational characteristics of such systems and evaluate the regulatory features. While many smaller problems can be studied and solved by conventional methods and computational facilities, such approach is inadequate for large-scale problems. For example: Global aspects of water and air pollution, modification of climate by environmental factors, mechanisms of diseases and mental disorders, etc. In order that these problems could be realistically studied and solved by computer simulation techniques, many new methods have to be introduced in terms of operational procedure as well as computer techniques and hardware. In addition, a new kind of institutional and organizational patterns have to be developed in

order to deal with multidisciplinary aspects of the problems (1,2). Here we consider only the problems for computer simulation and technical difficulties which are inherent in such an approach.

Once a problem is selected for computer simulation and model development, it is desirable to establish a basic procedure to deal with the problem. The following general steps can be proposed:

1. Assemble a competent multidisciplinary group of professionals to establish the framework for modeling.

2. Data collection from information storage and other sources.

3. Preliminary outline and scope of model systems.

4. Rough formulation of the problem and estimation of number of differential equations required.

5. Computer selection:

a. If suitable computers are available - selection has to be made in order to gear other steps with operational characteristics of the computer.

b. If suitable computers are not available via commercial sources, the only alternative is a program to develop a new computer and use this for computer simulation.

6. Develop detail model, formulate mathematically, perform computer simulation and compare model system performance with actual system.

7. Determine the limitations and inadequacies of the model and improve the model which includes further data collection. This step has to be repeated several times in order to obtain a realistic model system performance via computer simulation.

8. Once a realistic model has been established:

a. Study the dynamic performance of the system and carry out parametric analysis.

b. Determine extreme conditions where system becomes irreversibly uncontrollable and study the consequences; compare with the actual system in nature.

9. Recommend on the basis of computer simulation:

a. The permissible levels of various entities in the system and establish tolerance ranges.

b. Make recommendations for correcting the behavior of the system in nature.

c. Provide data for the enactment of new laws and environmental standards.

The computer problem and simulation. The facts are:

1. There is no computer on the market capable of solving large-scale dynamic systems in biology and environment related problems.

2. It is impractical to develop such computer by industry, since only a few of these are needed and operational characteristics are in a special category.

3. Only feasible way to overcome this problem is to carry out government supported special computer development and establish a few computer centers in the country.

4. Specifications for such computer design have to be established by a group of competent scientists:

a. Specialists from various problem areas.

b. Specialists in the computer field.

c. Specialists in administrative procedures for the computer utilization.

Who is going to make the decision for such computer development, and who is going to fund it?

The answer is: There is no government organization to do it, and there is no interdisciplinary competence in any government branch to take the initiative and carry out such steps! For such purpose, new institutions are required (2).

The following example will serve to focus the issue.

A practical model of a metabolic process is presented to illustrate the character of biological problems when analyzed in realistic terms. Tryptophan metabolism in pineal gland, which has recently been simulated on the computer, will serve as such an example. Fig. 1 shows the basic metabolic scheme and Table 1 provides the list of symbols and Table 2 the flow equations. Original paper should be consulted for problem formulation and computer simulation results. System entails 40 simultaneous differential equations.

The program scheme for analog computer is shown in Fig. 2. After an operational model system has been established on the computer, the time dependance and interrelationship of all functional entities can be studied. Technical details for organizing the problem on the computer can be found in another publication (4). Computer simulation reveals that such system can be stationary (Fig. 3) when regulatory compound (a_1) is absent, but will become oscillatory when it is present (Fig. 4 and 5). A more detailed relationship between various functional entities during the oscillatory cycle is presented in Fig. 6. A systematic study of dynamics of the model provides essential information in regard to basic biological functioning of the system.

One has to consider that living species and environment at large contain many oscillatory systems coupled together within the total aggregate system. For example, cyclic phenomena in reproductive physiology present indeed a formidable problem for computer simulation. However, regulation and control of reproduction of species can be adequately dealt with only when basic mechanisms of reproductive processes are understood. Needless to say, without a "new approach" there is little hope that large scale environmental problems and complex human diseases can be properly understood nor adequate solutions found.

REFERENCES

1. L. G. Wayne, Datamation, 17, 27, 1971.

2. R. Bowers, P. Hohenberg, G. Likens, W. Lynn, D. Nelkin and M. Nelkin, Am. Scientist, 59, 183, 1971.

3. F. Heinmets, Quantitative Cellular Biology, Marcel Dekker, Inc., N.Y., 1970.

4. F. Heinmets, Comput. Biol. Med. (in print).

TABLE 1

FUNCTIONAL ENTITIES AND SYMBOLS

E_0 - A group of enzymes induced by substrate s_6^1 or a_1

E_1 - Transport enzyme for tryptophan

E_2 - Tryptophan 5-hydroxylase

E_3 - Tryptophan 5-HTP decarboxylase

E_4 - Monoamine oxidase (MAO)

E_5 - Alcohol dehydrogenase

E_6 - Hydroxy indole methyl transferase (HIOMT)

E_5^3 - Aldehyde dehydrogenase

E_4^1 - Acetylating enzyme

E_c^1 - Adenyl cyclase

E_c^2 - Phospho diesterase

A_2 - Inhibitor of E_c^1

E_t - Transport enzyme

P_i - Internal pool

R - Inactive repressor

$\overset{*}{R}$ - Active repressor

G_1 - Genes

M - Messenger RNA

B - Ribosome

$\overset{*}{A}_1$ - Activator for M

A_1 - Inactive activator

H_1 - C_2 storage receptor (inactive)

$\overset{*}{H}_1$ - Active storage receptor

E_r - Norepinephrine release enzyme ($\overset{*}{E}_r$ - inactive form)

L - Light

C_p - Parenchymal cell storage receptor

C_n - Nerve cell storage receptor

E_o - Total enzymes synthesized as the result of a_1 promotion effect. (E_2, E_4^1 and A_2)

s_4^1 - Unbound s_4 in nerve cell

s_1 - External tryptophan

s_2 - Internal tryptophan

s_3 - 5-Hydroxy tryptophan (5-HTP)

s_4 - 5-Hydroxy tryptamine (5-HT)

s_5 - 5-Hydroxy indole acetate

s_2^1 - Tryptamine

s_6 - 5-Hydroxy tryptophol

s_7 - 5-Methoxy tryptophol

s_5^2 - 5-Methoxy serotonin

s_5^1 - N-acetyl serotonin

s_6^1 - Melatonin

s_6^3 - 5-Hydroxy indole acetic acid

s_7^3 - 5-Methoxy indole acetic acid

$[s_4]_p$ - External s_4 (pool)

$[s_6^1]_p$ - External s_6^1 (pool)

$[s_4C_p]$- s_4 stored in the parenchymal cell

$[s_4C_n]$- s_4 stored in the nerve cell

s_4^1 - Unbound s_4 in nerve cell

a_1 - Norepinephrine ($[a_1]_s$- stored

a_2 - E_c^2 inhibitors

i_m - Cycloheximide

i_n - Actinomycin D

C_1 - ATP

C_2 - CAMP ($\overset{*}{C}_2$bound form)

C_3 - AMP

i_1 - Seratonin storage transport inhibitors

i_2 -

i_3 -

i_4 - } various enzyme inhibitors

i_5 -

i_6 -

F. Heinmets, Comput. Biol. Med. (in print)

TABLE 2

FLOW EQUATIONS FOR THE MODEL SYSTEM

		Rate constant(k) subindex numbers
1.	$s_1 + E_1 \rightarrow s_2 + E_1$	1
2.	$s_2 + E_2 \rightarrow s_3 + E_2$	2
3.	$i_2 + E_2 \rightarrow [i_2E_2] \rightarrow E_2$	3,4*
4.	$s_3 + E_3 \rightarrow s_4 + E_3$	5
5.	$i_3 + E_3 \rightarrow [i_3E_3] \rightarrow E_3$	6,7*
6.	$s_4 + E_4 \rightarrow s_5 + E_4$	8
7.	$s_4 + E_4^1 \rightarrow s_5^1 + E_4^1$	9
8.	$s_4 + E_6 \rightarrow s_5^2 + E_6$	10*
9.	$s_4 \rightleftarrows [s_4]_p$	11, - 11
10.	$i_4 + E_4 \rightarrow [i_4E_4] \rightarrow E_4$	12, 13*
11.	$s_5 + E_5 \rightarrow s_6 + E_5$	14
12.	$i_5 + E_5 \rightarrow [i_5E_5] \rightarrow E_5$	15, 16*
13.	$s_6 + E_6 \rightarrow s_7 + E_6$	17
14.	$i_6 + E_6 \rightarrow [i_6E_6] \rightarrow E_6$	18, 19*
15.	$s_5 + E_5^3 \rightarrow s_6^3 + E_5^3$	20
16.	$s_6^3 + E_6 \rightarrow s_7^3 + E_6$	21
17.	$s_5^1 + E_6 \rightarrow [s_5^1E_6] \rightleftarrows s_6^1 + E_6$	22, 23, -23
18.	$s_6^1 \underset{\leftarrow}{\rightarrow} [s_6^1]_p$	24, -24

19. $s_4 + C_p \rightleftarrows [s_4 C_p]$	25, -25
20. $i_1 + C_p \rightarrow [i_1 C_p] \rightarrow C_p$	26, 27*
21. $s_4 + E_t \rightarrow s_4^1 + E_t$	28
22. $s_4^1 + C_n \rightleftarrows [s_4^1 C_n] \rightarrow [s_4]_p + C_n$	29, -29, 30
23. $s_4^1 \rightarrow X$	69
24. $i_1 + E_t \rightarrow [i_1 E_t] \rightarrow E_t$	31, 32*
25. $G_R + P_i \rightarrow G_R + R$	- *
26. $R + S_6^1 \rightleftarrows \overset{*}{R}$	33, -33*
27. $G_A + P_i \rightarrow G_A + A_1$	34
28. $G_A + \overset{*}{R} \rightleftarrows G_{*A}$	35, -35
29. $A_1 + \overset{*}{C}_2 \rightarrow \overset{*}{A}_1 + H_1$	37
30. $G_1 + P_i \rightarrow G_1 + M$	- *
31. $M + \overset{*}{A}_1 \rightarrow \overset{*}{M} \rightarrow M$	39, 38
32. $\overset{*}{M} + P_i \rightarrow E_o + \overset{*}{M}$	40
32A. $E_o \rightarrow X$	41
33. $\overset{*}{M} + i_m \rightarrow [i_m \overset{*}{M}]$	42*
34. $E_o \rightarrow E_2$	43
35. $E_o \rightarrow E_4^1$	44
36. $E_o \rightarrow A_2$	45
37. $a_1 + E_c^1 \rightarrow \overset{*}{E}_c^1 \rightarrow E_c^1$	46,47

38.	$A_2 + E_c^1 \rightarrow {}^*E_c^1 \rightarrow E_c^1$	48, 49
39.	$C_1 + {}^*E_c^1 \rightarrow C_2 + {}^*E_c^1$	53*
40.	$C_1 + E_c^1 \rightarrow C_2 + E_c^1$	51
41.	$C_1 + {}^*E_c^1 \rightarrow C_2 + {}^*E_c^1$	52
42.	$C_2 + E_c^2 \rightarrow C_3 + E_c^2$	78*
43.	$A_2 + E_c^2 \rightarrow {}^*E_c^2 \rightarrow E_c^2$	79, 80*
44a.	$a_1 + H_1 \rightarrow {}^*H_1 \rightarrow H_1$	56, 55
44b.	${}^*H_1 + C_2 \rightarrow {}^*C_2 \rightarrow {}^*H_1$	54, 50
45.	$s_7 \rightarrow X$	57
46.	$s_7^3 \rightarrow X$	58
47.	$s_5^2 \rightarrow X$	59*
48.	$C_3 \rightarrow X$	60*
49.	$A_2 \rightarrow X$	63
50.	$C_2 \rightarrow X$	64
51.	$s_6^1 \rightarrow X$	65
52.	${}^*A_1 \rightarrow X$	66
53.	$[s_4]_p \rightarrow X$	67
54.	$[s_6^1] \rightarrow X$	68
55.	$E_2 \rightarrow X$	70
56.	$E_4^1 \rightarrow X$	71
57.	$[a_1]_s + E_r \rightarrow a_1$	72*
58.	$E_r + L \rightleftarrows {}^*E_r$	73, -73*

59.	$s_2 + E_3 \rightarrow E_3 + s_2^1$	74
60.	$s_2^1 \rightarrow X$	75
61.	$s_5^1 \rightarrow X$	76
62.	$A_1 \rightarrow X$	77

Note: Equations indicated by star (*) are not included into differential equations. However, their role is simulated on the analog computer by manipulation of appropriate rate constants. Details will be discussed in the text.

F. Heinmets, Comput. Biol. Med. (in print).

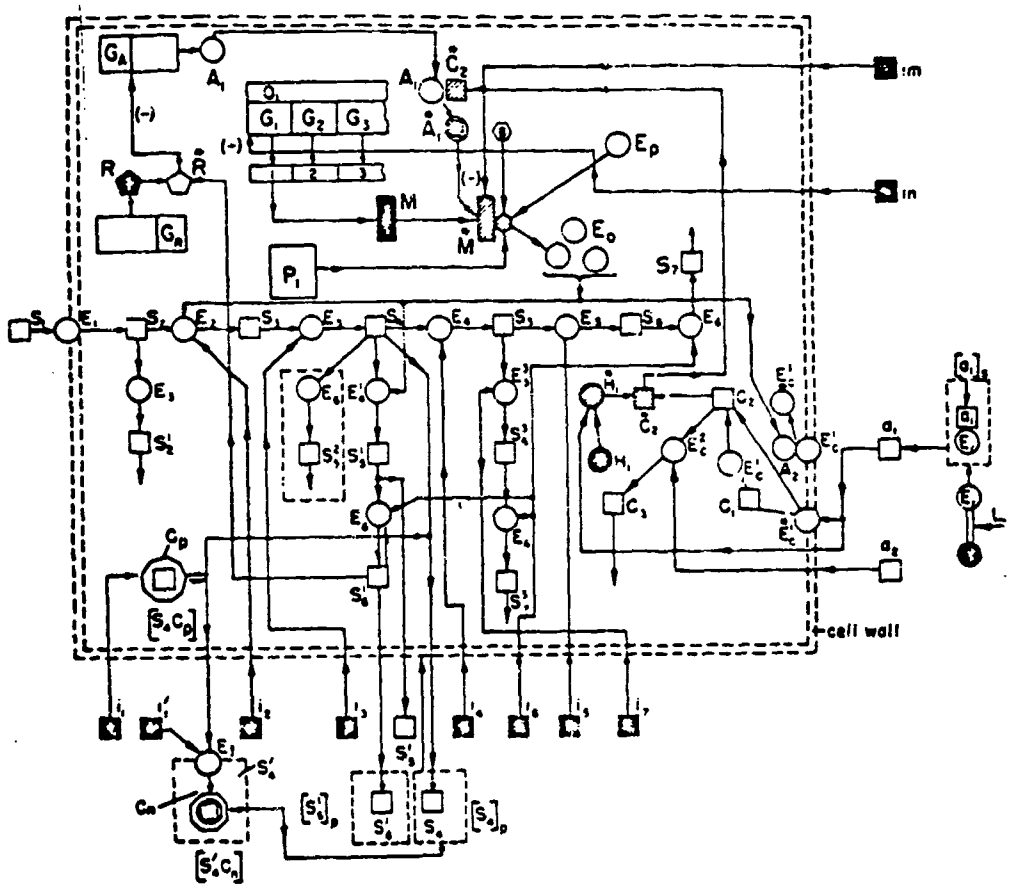


Figure 1

A schematic model for tryptophan metabolism in the pineal gland (4).

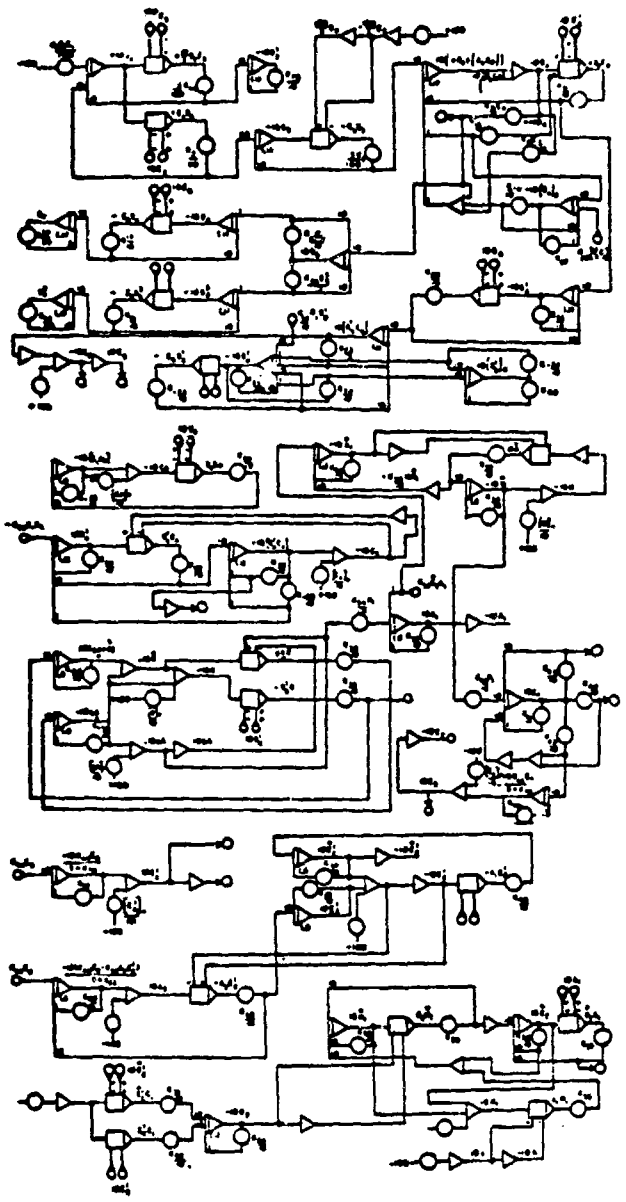


Figure 2

Analog computer program for the model (4).

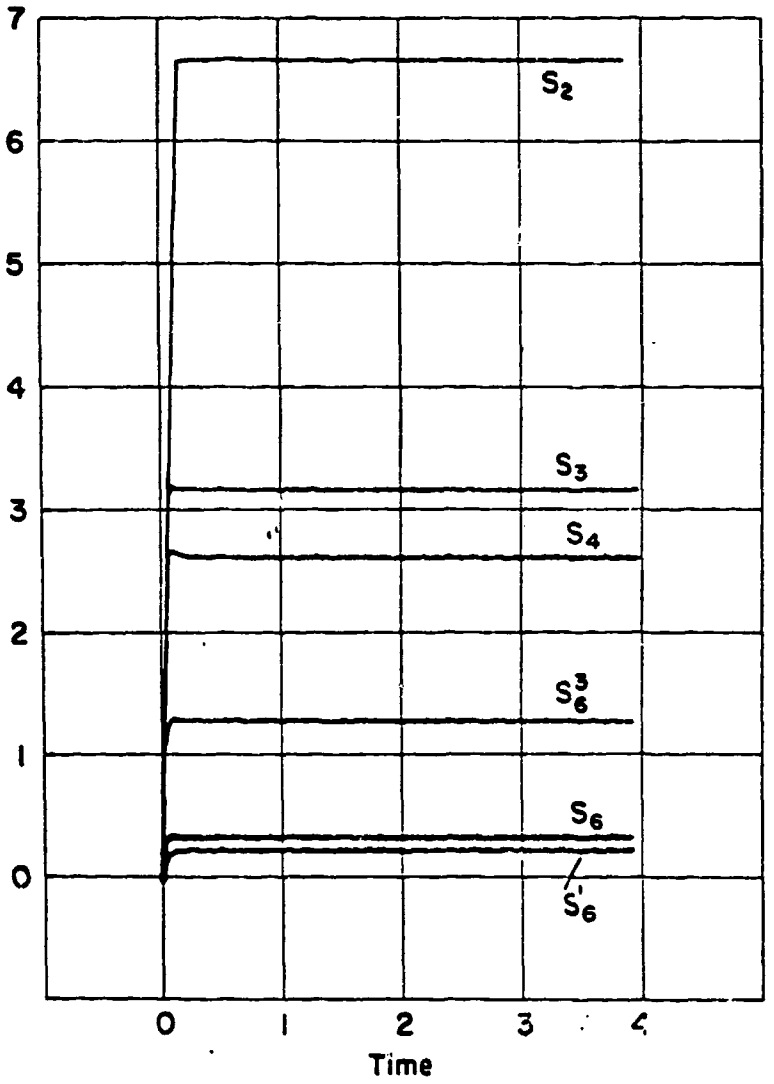


Figure 3

Stationary concentration levels of various substrates in the system where $a_1 = 0$ (4).

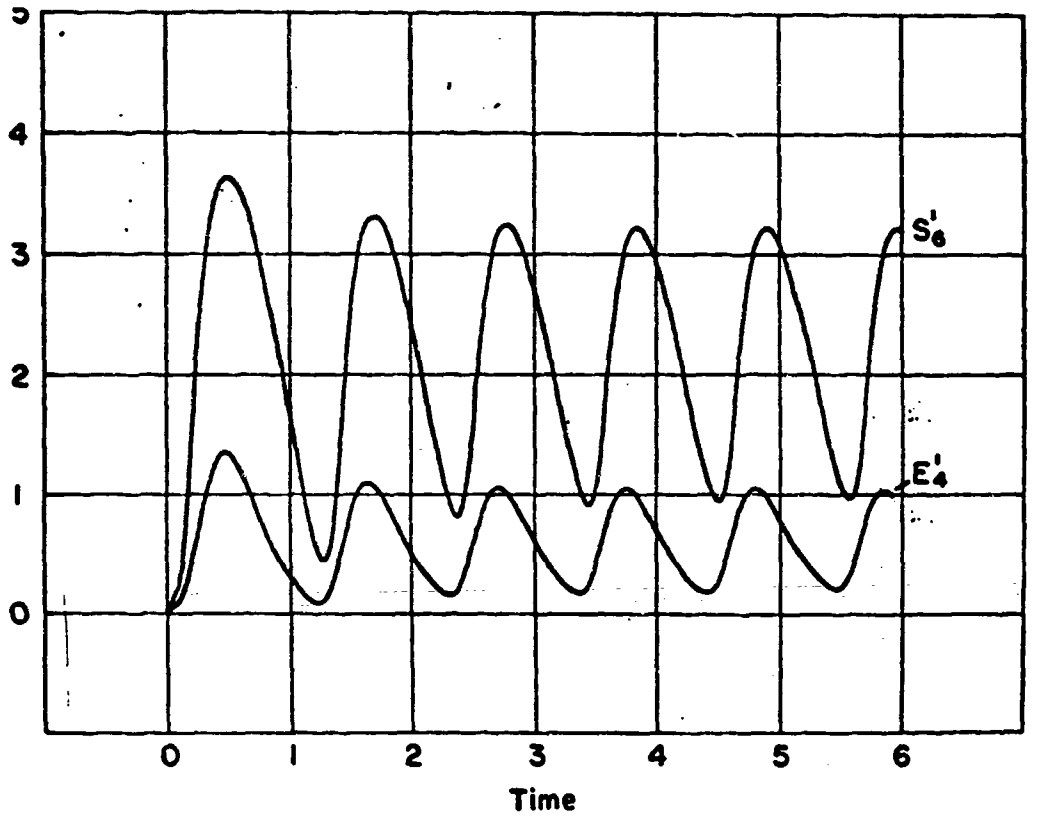
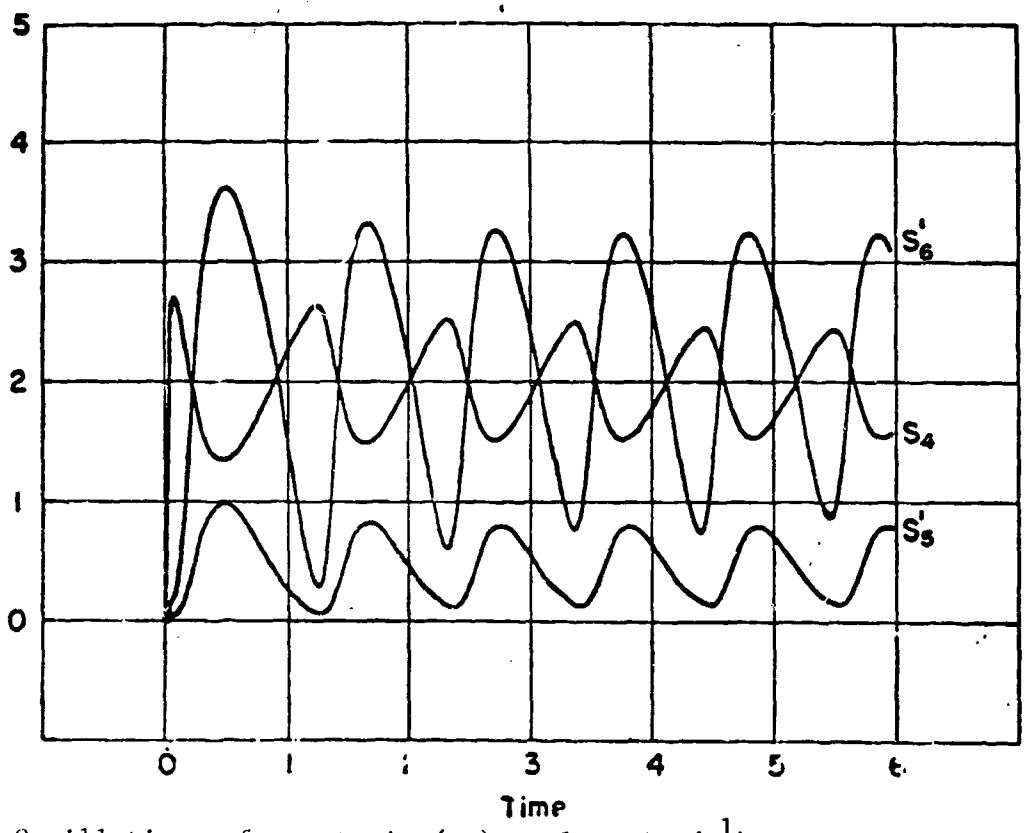


Figure 4

Oscillatory relation between melatonin (s_6^1) and seratonin acetylating enzyme (E_4^1) (4).

Figure 5



Oscillations of seratonin (s_4), melatonin (s_6^1) and N-acetyl-seratonin (s_5^1) as function of time (4).

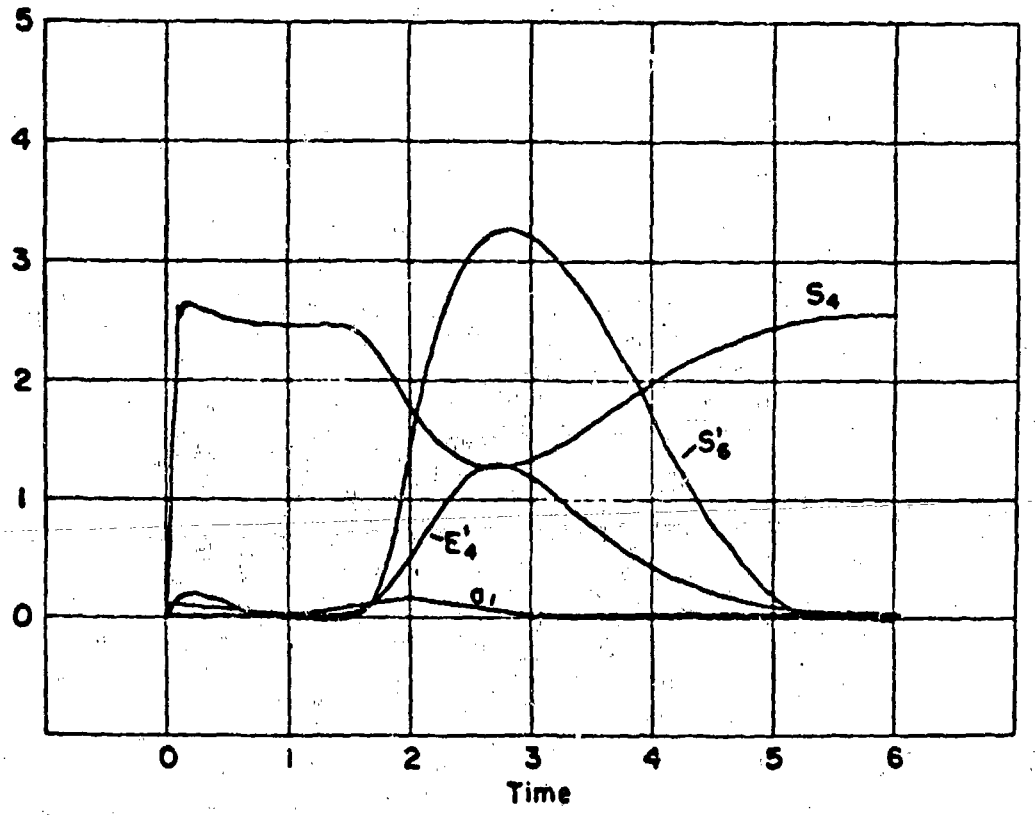


Figure 6

Induction of oscillatory behavior of metabolic system by transient pulse of NE(a_1). Enzyme E_4^1 and substrates s_6^1 and s_4 are recorded (4).

THE JOHNS HOPKINS UNIVERSITY
APPLIED PHYSICS LABORATORY
SILVER SPRING MARYLAND

THE USE OF MULTIPLE REGRESSION EQUATIONS IN THE CAMPAIGN EXECUTION MODEL

Sol Haberman and Norman T. Rasmussen

ABSTRACT. Ships in task force operating areas are subjected to air attacks. For all such attacks a basic regression equation type is used which represents a family of least squares fits to the best available data. Three kinds of data are used as independent variables; the effective number of the various types of resources in the attacked force, the quality of the defender's missiles, and the number of attackers. The equation is used repetitively with different sets of prestored coefficients and powers in order to predict the mean number of hits on units of both sides. These hits are both calculated and accumulated in order to record damage levels individually by hulls for specific units of the various types. Deletion of units occurs when input thresholds of damage (number of hits for this type) have been reached. An input criterion determines the vulnerability of units to damage by surviving hulls.

These calculations have been organized so that their logic may be applied to all forms of attack and damage, not only to the immediate air warfare requirement.

I. INTRODUCTION. The Campaign Execution Model is a computer program for simulation of a major war at sea. It was originally conceived by Arthur W. Pennington and Jerome Bracken.

The purpose of this paper is to discuss some of the recent developments in the Campaign Execution Model portion of the procedure, the whole of which is known as the IDA Campaign and Allocation Model.*

The CEM which is the focus of this discussion is designed to deal with a wide range of scenarios representing threats, resources, and strategic objectives within the context of a major conventional war in support of current strategy. Its specific detailed function is to do bookkeeping; to keep track of resources, force interactions, resource attrition, and military accomplishments within the scenario framework by the user. Data representing force effectiveness and encounter rates which are CEM inputs are obtained from external analyses. The value of a given outcome is determined by the quality of these external analyses.

---

*The IDA Campaign and Allocation Model, of which the CEM is a part, is the Sequential Untrained Minimization Technique (SUMT), was assigned to the Planning Analysis Group of APL/JHU by the Director, Systems Analysis Division (OP-96) in April 1969.

THE JOHNS HOPKINS UNIVERSITY
APPLIED PHYSICS LABORATORY
SILVER SPRING MARYLAND

The subject matter of this paper is to report on the new representation of the air attack on ships making up the various task forces.

The revised CEM model has been tested and used on both the CDC-1604 and IBM 360/91 computers. The program is written in FORTRAN IV.

The procedures which led to the major revision of the air attack had their origin with some modest initial changes in the antisubmarine portions dealing with detection probabilities.

Historically, the initial studies of the CEM logic indicated a need to change the method of calculating the probabilities of detections of RED submarines by task forces, convoys, amphibious assault groups (AMP) and underway replenishment groups (URG). The procedure finally adopted was one which utilizes curves to represent detection probabilities for a range of BLUE and RED types and numbers of forces. The curves were derived from data developed for the ASW Force Level Study.

The ASW Force Level Study source data was available in the form of probabilities of detection of a single RED submarine where each probability was a function of the range between the center of the RED force and the center of the BLUE force. Probability estimates were given for varying numbers of "Poor" (SQS-23) and "Good" (SQS-26, SQQ-23) BLUE escorts and for varying numbers of RED nuclear and conventional submarines. The probabilities combined in four ways, "Poor" vs. Nuclear, "Poor" vs. Conventional, "Good" vs. Nuclear and "Good" vs. Conventional and were restated in the form of curves, using least squares as shown schematically in Table I. More recent information indicates that these curves are not entirely adequate at the lower end. The curves give probabilities of detection which are always zero for all types of forces when no escorts are present. With operational and cost limitations on the numbers of escorts which can be assigned, the curves cannot and do not in practice reach probabilities of 1.00 at the other end.

Since many interactions involve mixes of units on both sides, interpolations among the set of four curves become necessary. These interpolations give the same results whether they are performed first between RED submarine types (Nuclear vs. Conventional) or between BLUE escorts (Poor or Good).

The expression used for the detection of a Red submarine is:

$$\text{Probability} = \frac{PN(P)(N) + GN(G)(N) + PC(P)(C) + GC(G)(C)}{(P + G)\ (N + C)}$$

where P = number of Poor BLUE Escorts

G = number of Good BLUE Escorts

N = number of RED Nuclear Submarines

C = number of RED Conventional Submarines

and PN = Prob. of Detection of RED Sub (Given Poor Escorts vs. RED Nucs)

PC = Prob. of Detection of RED Sub (Given Poor Escorts vs. RED Conventionals)

GN = Prob. of Detection of RED Sub (Given Good Escorts vs. RED Nucs)

GC = Prob. of Detection of RED Sub (Given Good Escorts vs. RED Conventionals)

Probabilities of Detection of a Single Red Submarine

Task Forces (Subroutine "WARCTF")

$$PN = K_{PN} * \sqrt{E}$$

$$PC = K_{PC} * \sqrt{E}$$

$$GN = K_{GN} * \sqrt{E}$$

$$GC = K_{GC} * \sqrt{E}$$

PN = Poor Sonar vs. Nuclear Submarine

PC = Poor Sonar vs. Conventional Submarine

GN = Good Sonar vs. Nuclear Submarine

GC = Good Sonar vs. Conventional Submarine

K = Numerical constants developed by least squares calculations

E = Number of Blue Escorts

TABLE I

Note: The equations for probabilities of detection used by Amphibious Forces, Underway Replenishment groups and Convoys are the same as above. The differences among these forces are represented by K values.

II. BACKGROUND OF AAW SUBROUTINE REVISION. Major effort was directed to developing new concepts for and rebuilding the AAW subroutines known as TASKAIR of the CEM. This set of subroutines is vital to the operation of the CEM because in it is conducted the entire RED air and cruise missile attack against the BLUE carrier task force. The interactions in TASKAIR can have a considerable effect on BLUE's conduct and success in the war.

The model used for the AAW study was the Systems Interaction Model II (SIM II) which was developed by APL/JHU and which has been used extensively by the Navy (primarily the assistant DCNO for War Gaming matters (OP-06c)) for about six years. The SIM II model is a Monte Carlo event-store simulation of naval AAW interactions.

The AAW study conducted with SIM II was not intended to be an all-inclusive study of the AAW problem. The primary objective was to study the various elements which affect the outcome of an AAW conflict in order to determine within certain limits which are the important variables and, determining this, to develop an improved flexible methodology for representing AAW interactions in the CEM. The second purpose for using SIM II was to obtain a reasonable AAW data base with which real studies could be made.

The initial intent with respect to the use of this model was to obtain data from which to derive an equation for predicting the number of RED aircraft shot down and to incorporate this into the CEM. As runs were being made it became evident that there were many additional types of information available from the SIM II outputs which could also be used in the CEM. An example of this is number of hits on ship types such as CV's, DLG's, DDG's, and DD's. It also became clear that the SIM II study would produce information which could be expressed in predictive equations representing the effectiveness of the various BLUE weapon systems. These, while not immediately applicable to CEM, have usefulness elsewhere.

In order to obtain the representative equations required to summarize a typical range of RED and BLUE antiair warfare interactions, SIM II was used to simulate a variety of attack carrier task force dispositions which were subjected to RED air attack.*

Red raids varied in size from 30 to 150 bombers in steps of 15 and cruise missile attacks varied in size from 60 to 300 missiles in steps of 30. Following the principle of maximum concentration of force, each bomber raid consisted of five equal segments approaching the task force in a 90^{0} arc with all reaching ASM launch point at approximately the same time.

---

*Further details on these interactions are available in the Naval Force Methodology Study, APL/JHU/PAG No. 40-71, CNO/OP-96/TR-3300, on pages 13-29.

Two basic categories of ships were used in 102 BLUE dispositions. The first consisted of those types that contributed directly to the AAW defense of the force by having the capability to shoot down RED bombers and/or missiles. The second category of BLUE ships were those that could not shoot down RED bombers and missiles, but which did contribute indirectly to the outcome of the battle by acting as additional targets for RED missiles to acquire and attack, thereby reducing the number of hits taken by the CVA, CG, DLG, and DDG types. The ships in this category included the CVS and DD's for simulated ASW protection and, at times, an Underway Replenishment Group (URG) and an amphibious force (AMP) each with its accompanying ASW escorts.

The data summarized from SIM II* requires multiple regression analysis because no single input variable by itself is adequate to explain outcomes such as bombers shot down, hits on the CVA's etc. There are eight basic driving variables used to predict these outcomes: i.e. numbers of CVA's, CG's, DLG's, DDG's ..., number of attackers and weapon performance level. Each game was analyzed in two ways: (1) as an aircraft attack using ASM's, (2) as a cruise missile attack treating the ASM's successfully launched as the attacking cruise missiles.

Each of 1,017 different games was repeated five times and mean numbers were obtained for 47 different outcomes for the aircraft attack and 34 different outcomes for the cruise missile attack.

III. THE NEW TASKAIR SUBROUTINES. The TASKAIR purpose remained the same after all changes had been completed; to subject the BLUE ships in each Task Force Operating Area to RED air attack in each ocean. RED attack is by sub-launched cruise missile, by stand-off aircraft launched missiles, or by coordinated or uncoordinated combinations of these two. Inputs control their order and coordination.

For all such attacks, a single basic equation type is used to represent a least squares fit to the best available data as a means of predicting the various results of the attack. The actual form of the equation is that of the multiple-regression procedure to be described. The equation uses three kinds of data as independent variables: the effective number of the various types of BLUE resources in the attacked force, the quality of the BLUE SAM missiles, and the number of RED attackers. The equation is used repetitively with different sets of prestored coefficients and powers in order to predict the mean number of hits on BLUE and RED units. These hits are both calculated and accumulated in order to record damage levels individually by hulls for the CVA's, DLG's, DDG's, the CVS and the CG, and by class type for all other units. Deletion of units occurs when input thresholds of damage (number of hits for this type) have been reached. Also provided is an input criterion that determines the vulnerability of units to damage by surviving hulls on a "least-hits-to-the-most-damaged-unit" basis, or a most-to-least basis, or in some weighted combination of these.

*The multiple regression procedure used here for least squares curve fitting to the SIM II data, is the BCC Library Routine No. 8.06.01, adapted by J. Bramhall to the 7094 at APL. This follows a computational method developed by M. A. Efroymson and is discussed in Chapter 6 of "Applied Regression Analysis" by Draper and Smith (Wiley 1966).

These calculations have been organized so that their logic may be applied to all forms of attack and damage, not only to the immediate air warfare requirement. Much of this procedure is now in use with RED submarine torpedo attacks.

Thus, TASKAIR performs five major functions:

1. Determines the total number of hits on all units, either individually, or by unit type, by referencing appropriate equations.

2. Combines these hits with past damage records.

3. Deletes "killed" units (i.e. units for which the total damage exceeds an input kill threshold for the given type of unit).

4. Reflects intelligence and BLUE doctrine by causing the assignment of hits in subsequent attacks on individual surviving ships of the CVA, DLG and DDG type to be inflicted in a "least-hits-to-most-damaged-unit" order, "most-to-least" order, or a weighted combination of these.

5. Uses the number of hulls present for the CV's, DLG's and the CG's so as to provide an expansion of the performance criteria studied to include the number of hull carrier-days on the line, CG days on the line, etc.

IV. DETERMINATION OF HITS. A schematic example of the multiple regression equation used to predict the mean number of hits on the CVA which gets hit worst, or second worst, etc. by a RED aircraft attack is:

$$\text{HITS} = C_0 + C_1 * (CV)^{P_1} + C_2 * (CG)^{P_2} +$$

$$C_3 *(DLG)^{P_3} + C_4*(DDG)^{P_4} + C_5*(DD)^{P_5} + C_6*(A+U)^{P_6} + C_7*(A/C)^{P_7} + C_8(QSAM)^{P_8}.$$

Where,

the C's are the set of real number coefficients, and the P's are the set of positive real numbers uniquely associated with this result. The values for the C's and P's are determined by the least squares fit to the best data available.

The quantities in the parentheses are the independent variables of the equation. Their values are:

CV = effective number** of CVA's in the task force + effective number of CVS's in the task force.
CG = effective number of CG's in the task force.
DLG = effective number of DLG's in the task force.
DDG = effective number of DDG's in the task force.
DD = effective number of DD's in the task force.
A+U = effective total number of units of all types in the AMP and/or URG force associated with the main task force.
A/C = number of attacking RED aircraft (or, number of attacking cruise missiles, as required).
QSAM = quality level of BLUE SAM's used to shoot down RED aircraft or missiles. The input value represents high performance, low performance, or an interpolated level of performance.

The equation used to predict the mean number of hits on some resource other than a CVA is identical in form to the above equation. In this event, the program selects from a series of "RED aircraft attack" arrays, those values of the coefficients and powers pertaining to the result required.

The application of the equation to a cruise missile attack requires only a minor variation in the list of independent variables: in the schematic example above, "A/C" becomes the number of attacking cruise missiles. The program now selects the appropriate constants from a series of "cruise missile attack arrays."

For a given air attack by RED, whether by aircraft or by cruise missile, TASKAIR works through the sequence of equations to get the individual mean number of hits (from the unit with the largest number of hits to the unit with the smallest within types) on the various CVA's, DLG's, DDG's, the CVS and the CG. All other units are represented by a single prediction equation for each unit type. Specifically, the DD's attached to the carrier group, the AMP and URG main-body ships, and the AMP and URG escorts each receive their hits in totals for their respective types. These particular units are treated as aggregates because the number of hits which kill each unit generally is very small.

**By effective number of units is meant the total obtained by adding together the effectiveness values of the surviving units of a given type. User inputs determine the fractional effectiveness of a unit after it has received one or more hits. Although the provisional relationship between hits and unit effectiveness is linear, this can be modified to reflect cases where earlier or later hits have disproportionate effects.

An example of the current procedure is: given a kill deletion criterion of 6 hits, a unit receiving 1 hit would have 5/6 effectiveness and with 2 hits would have 4/6 effectiveness.

V. COMBINING CURRENT HITS WITH PAST DAMAGE RECORDS. For each individual or group result obtained from an equation, there exists a matching cumulative damage record of hit counter. The content of this hit counter is the damage from prior air and torpedo attacks. These cumulative hit counters are permanent records associated with each of the CVA's, DLG's, DDG's, the CVS and the CG; for the other units, hit records are kept by totals for all units of a given type. A detailed example of an individual hit counter is given under the heading "Assignment of Hits in Subsequent Attacks."

VI. DELETING UNITS. The third function of TASKAIR is to test up-dated damage or hit counters for kills.

In the case of hits on the grouped units present, the number killed is determined by the number of times the hits-to-kill criterion divides into damage or hits calculated for this type of unit. For example, if there are seven hits on a number of DD's with the carrier task force, and if it takes two hits to kill a DD, then 3.5 DD's are deleted.

If the numbers of hits on a unit treated individually (CVA, CVS, or DDG) equals or exceeds the input hits-to-kill criterion for this type, the unit is deleted.

Thus for both types, the effective number (not hulls) of a given type of unit remaining after an attack equals the original effective number prior to the current attack minus effective units deleted with this attack. However, group and individually treated units do differ in regard to the method of "hull" deletion.

VII. ASSIGNMENT OF HITS IN SUBSEQUENT ATTACKS. The cumulative damage records for the groups of non-deleted individual units (CVA's, DLG's, and DDG's) are ranked in order to reflect the damage doctrine in effect. The damage doctrine input reflects the degree of BLUE's ability to align his resources optimally with respect to the azimuth of the next air attack, or, or RED's ability to thwart BLUE's predictive capability. Consequently, the input choice causes the damage from the RED air attack to be suffered according to one of three options:

1. The units receiving the most hits are the units with the least amount of prior damage, or
2. The units receiving the most hits are the units with the most amount of prior damage, or
3. A weighted selection of options (1) and (2).

As an example assume that three surviving CVA's have received totals of 2.0, 2.5, and 3.9 hits respectively in all previous attacks. Further, assume that the three have received 1.1, 1.7, and 2.4 hits in the current interval. The individual cumulative damage or hit counters would be tallied differently under different damage doctrines although the total number of hits is identical:

CUMULATIVE DAMAGE (HIT) COUNTER CONTENTS

	Damage Option 1			Damage Option 2		
	Old	New	Total	Old	New	Total
1st CVA	2.0	2.4	4.4	3.9	2.4	6.3
2nd CVA	2.5	1.7	4.2	2.5	1.7	4.2
3rd CVA	3.9	1.1	5.0	2.0	1.1	3.1
	8.4	5.2	13.6	8.4	5.2	13.6

(Throughout these calculations outcomes representing subtotals are adjusted in terms of predicted totals. For example, it may be assumed here that in these calculations the predicted hits on all of the CVA's 8.4 and 5.2, have been adjusted to conform to fleet totals in each of the go-rounds and that hits on individual CVA's have in turn been adjusted to conform to CVA totals. This is reasonable since fitting curves to outcomes representing large aggregates of units is more reliable than to smaller aggregates.)

Note that there options may produce different numbers of kills depending upon input kill criteria. E.g., if the kill criterion is 6.0, no kill results under option 1 and 1 kill result under option 2.

VIII. THE "HULL" COUNTING CONCEPT. Historically, CEM and other models have measured results in terms of the effective number of units. A new concept has been added here by measuring results in terms of the gross number of units, or "hulls". Both methods are important and they complement each other.

The concept of hull counting can be used in at least two ways: (1) to simulate internally in the model more realistic interactions in terms of fuel and ammunition requirements, and (2) as a measure of effectiveness, specifically "CVA Hull Days on the Line", which statistic is now available as an output.

The new TASKAIR methodology demonstrates that the experience obtained from fleet exercises, from running computer models and from theoretical investigations can be reduced to equations referenced as needed in CEM.

The new TASKAIR subroutines show that by using predictive equations developed from a reliable data source both the effects of attacks on units in the task force areas and the losses to the attackers can be determined and used. The subroutines calculate and maintain damage records, delete units and reflect the redisposition of force to maximize defensive potential.

Structurally the set of small TASKAIR subroutines deals with actions such as hits, kills, and damage calculations. These action subroutines are generally applicable to all types of warfare. For example, the methodology of the hits-to-kill calculations and the updating of damage histories has been applied to the torpedo warfare of the WARCTF subroutine of CEM.

IX. SUMMARY OF METHODOLOGY. The use of SIM II and the study of data generated by the model has led to the development of two new aspects of the force structure methodology which have been incorporated in the CEM. Both of these new developments can, and have, been applied equally well to ASW and AAW problems. These developments are:

1. The results of other investigations, (whether they be obtained from theory, from fleet exercises, or from using models such as SIM II) can be represented by equations used in subroutines by the Campaign Execution Model as required.

2. The CEM Model can remove units from the action according to input damage levels fixed by input choice. The outcomes of SIM II were summarized, not only in equations which predict mean number of total hits on all CV's, all DLG's, and on all DDG's, but in the form of predicting a mean number of hits on the CV which gets hit worst, on the CV which gets hit next worst and so on through the list for all of the types of units of interest.

In addition to the equation originally sought, giving the number of attacking RED aircraft shot down, there are now two basic groups of equations obtained from running SIM II based on the type of RED air attack. The first group represents attacks by RED aircraft in which the number of aircraft in the raid is one of the inputs. The other group represents attacks by cruise missiles from RED submarines in which the number of cruise missiles launched is an input.

The equations predicting hits on task force, underway replenishment groups, and amphibious units provide the framework of the new subroutines. The subroutines have been programmed so that a single basic equation is used repetitively (cycling through both types of air attack) with appropriate coefficients and powers utilized for each kind of BLUE and RED resources to predict mean numbers of hits on the BLUE units of interest. These predicted hits are added to cumulative hit records which are maintained throughout a run. Units are deleted according to input thresholds using these records.

These developments have been incorporated in the new TASKAIR subroutines and in the subroutines which calculate damage from submarine torpedo attack.

REFERENCES

1. N. R. Draper and H. Smith, "Applied Regression Analysis", Wiley, 1966.

2. S. Haberman and J. Heidt, "Naval Force Methodology Study", APL/JHU/PAG No. 40-71, CNO/OP-96/TR-3300.

DESIGN OF SURVEY SYSTEMS USING NONLINEAR PROGRAMMING METHODS

CPT Clifford W. Greve
Advanced Technology Division
Computer Sciences Laboratory
US Army Engineer Topographic Laboratories
Fort Belvoir, Virginia

ABSTRACT. This paper discusses theory and results of an attempt to use nonlinear programming methods to arrive at an optimal, in the sense of least cost, solution to the design of a survey system to meet specified accuracy. In other words, the method determines the combination of various types of observations which will yield the required accuracy of control points for a minimum cost. The theoretical background of the procedure is discussed, and methods of extension to photogrammetry and other sciences are presented. Much of the paper is concerned with discussing results of numerical solutions for the optimal design of several small, but typical mapping problems.

It is believed that this research is original with the author, as extensive literature searches and correspondence has produced no knowledge of prior research into this application of nonlinear programming. The method at its current state of development appears to be capable of yielding significant improvements to the present concepts of survey network design.

DISCUSSION OF PROBLEM. In experimental science a frequently recurring problem is to determine certain non-measurable parameters which are functionally related to measurable parameters. In addition, in most cases the relating functions are known. If one now knew the influence of cost upon the value of the variance-covariance matrix of the observable parameters, then, by the method of least squares he could arrive at the influence of cost upon the accuracy of the non-measurable parameters. One may then reverse the problem and ask what would be the minimum cost of attaining a specified accuracy for the non-observable variables. This is the problem toward which this paper is addressed. It should be noted at this point that the mathematics and method presented herein are applicable to all problems of optimization of experimental design as outlined above. At any point where a method is restricted to the specific problems of geodesy and photogrammetry, particularly horizontal control surveys, special mention will be made of this fact.

At this point a brief review of the history of attempts of solution of the above problem might be in order. The first such attempts consisted of generalized specifications. These specifications were generally rigid enough to insure that accuracy requirements were met, but often required far more work than actually required.

The remainder of this paper has been reproduced photographically from the author's manuscript.

With electronic computers came the age of computer simulation. By choosing the set of measurements to be made before actually making them, the error could be propagated and the error for the non-measurable parameters found. One would then pick a near optimal solution simply by choosing the least costly set of observations from among those sets of observations yielding acceptable results for the unknowns. With the advent of the "Kalman Filtering" algorithms, which are nothing more than sequential least squares if no time variation is involved, the capability for easily adding and deleting observations was realized. This facilitated the search for least cost solutions. However, since for most real world problems, there is a practical infinity of possible observation sets, the distinct possibility still exists that the optimum (least cost) data set might never be tried. Thus, although a minimum cost for the set of solutions tried can be obtained, the solution giving true minimum cost may not be in the set tried, and therefore will not be discovered.

The method proposed herein alleviates this problem of omission of data sets, because all possible observations are considered, and the computer picks which ones to use to arrive at an optimum configuration. Thus, the solution of the design of experiments problem becomes less dependent upon human past experience, and more dependent upon analytical methods.

Summary of Results from Error Theory

Since results for minimum variance and maximum likelihood estimators for observation equations of the form $AX = L+V$; where A is known coefficient matrix, X is vector of parameters, L is vector of observations with variance-covariance matrix $\sum_{LL}$, and V is vector of residuals with $E(V) = 0$; are well documented in the literature (3), only a brief summary of pertinent formulae will be presented here.

If the mathematical model may be represented as $AX = L+V$, then the variance-covariance matrix $\sum_{XX}$ of the least squares estimate of X (minimum variance if $f(X) = 0$ and variance of X finite, also maximum likelihood if L normally distributed) is $(A^T \sum_{LL}^{-1} A)^{-1}$. Although this is all the statistical knowledge which is needed for the method, it would be wise for anyone not familiar with statistical methods to study some other references before continuing to read the remainder of this paper.

Discussion of Linear and Nonlinear Programming

A mathematical programming problem is defined as any problem requiring the maximization or minimization of a function subject to equality or inequality constraints. Symbolically the problem looks like

$$\text{max or min} \qquad f(X)$$

subject to

$$\begin{matrix} g_1(X) \\ \vdots \\ g_n(X) \end{matrix} \begin{matrix} \geqq \\ \vdots \\ \leqq \end{matrix} \begin{matrix} b_1 \\ \vdots \\ b_n \end{matrix}$$

where $\begin{matrix} \geqq \\ \leqq \end{matrix}$ means that one of the symbols $\geqq$, $\leqq$, $=$ applies, and X is a vector. The g_i and f may be any functions of X, linear or nonlinear. If they are all linear, the problem is termed a linear programming problem. Otherwise it is a nonlinear programming problem.

A practical method for solving the linear programming problem was developed by George Dantzig in 1948. This algorithm, called the simplex algorithm, consists of an ordered search of the vertices of the feasible solution set (the set of all vectors X which satisfy the constraints). The search is ordered such that the next vertex looked at is always at least as good (from the viewpoint of maximizing or minimizing the function f) as the one before. Since for the linear programming problem, the boundaries of the feasible solution sets are hyperplanes in m space (if X is an n vector), then there are only a finite number of such vertices, and therefore the process must converge in a finite number of iterations. This still leaves open the possibility that, for large m and n, the number of iterations could become a very large finite number, but in practice it has been found that the algorithm will normally converge in a very reasonable number of iterations.

A mathematical description of the simplex algorithm follows. No proofs are given, or even hinted at. Proofs may be found in (1), or any good text on linear programming.

Given the linear programming problem

$$\min \sum c_j x_j \text{ or } C^T X$$

subject to

$$\begin{aligned} \sum a_{1j} x_j &\leq b_1 \\ &\vdots \\ \sum a_{ij} x_j &\leq b_i \\ \\ \sum a_{i+1,j} x_j &= b_{i+1} \\ &\vdots \\ \sum a_{nj} x_j &= b_n \end{aligned}$$

We change the problem into so called standard form by adding so called slack variables to each of the first j constraints to transform them into equalities. The cost of these slack variables is, of course, zero, so the vector C will not be influenced.

We now have

$$\min C^T X$$

subject to

$$
\begin{aligned}
\sum g_{ij} X_j + X_{n+1} &= b_i \\
&\vdots \\
\sum g_{ij} X_j + X_{n+i} &= b_i \\
\sum g_{i+1,j} X_j + 0 &= b_{i+1} \\
&\vdots \\
\sum g_{nj} X_j + 0 &= b_n
\end{aligned}
$$

The above is called the linear programming problem in standard form.

Changing the notation slightly, we can write the above problem as

$$\min C^T X$$

subject to

$$GX = B \qquad \text{where G is a matrix.}$$

For the problem above, we define a basic feasible solution as any solution X having n positive elements. (For any problem, either n will be less than m or one or more of the constraints will be redundant.) This is not a precise definition, but it is as close as one can come without assuming a detailed knowledge of convex set and linear manifold theory. The interested reader will find this elaborated upon in (1).

We will now define the simplex algorithm beginning with the assumption that we possess one basic feasible solution $\hat{X}$. Define C^* to be the costs associated with the current basis vectors in order. Define the vector $Z = C^{*T}G$. Form $Z^* = C^T - Z$. There are three possible cases to consider.

1. If $Z_k^* \geq 0$ for all k, then $\hat{X}$ is optimal.

2. If for some k (not corresponding to one of the positive elements in X) $Z^* < 0$ and $g_{ik} > 0$ for some i, then by replacement of a vector in the basic feasible solution, a better value of the objective function may be obtained. The vector to come into the basis is chosen by the minimum of all the Z^*, Z_k^* satisfying the above condition, and the vector to leave the basis is chosen as the minimum over j of $\frac{b_j}{g_{jk}}$, subscript called r.

This is accomplished by setting the new $g'_{ij} = g_{ij} - \frac{g_{ik}}{g_{rk}} g_{rj}$ if $i \neq r$, and

$$g'_{rj} = \frac{g_{rj}}{g_{rk}}, \quad b'_{ij} = b_{ij} - \frac{g_{ik}}{g_{rk}} b_{ij}, \quad i \neq r \text{ and } b'_{rj} = \frac{b_{rj}}{a_{rk}}.$$

One should note that this change of basis vectors is exactly the same as the process involved in Gaussian Elimination. In fact, Gaussian Elimination is simply a process of changing basis vectors, although it is seldom presented as such.

3. If for some k $Z_k^* < 0$ and $g_{ik} \leq 0$ for all i, $i = 1,\ldots,p$ then there exists no lower bound for Z and the problem has no optimal solution.

The iterative procedure is then continued with the new basis until either an optimum solution is found, or the non-existence of a minimum is determined.

We have assumed that the user has available an initial feasible solution. In very few problems will this be the case. Therefore, the user will be faced with determining a starting point for the simplex algorithm. A method called the method of artificial variables has been developed to solve this problem. The method consists of appending to the right of the G matrix of the linear programming problem in standard form a unit matrix. The cost function is changed so that the costs of the actual variables are zero, and the costs of the new variables introduced, called artificial variables, is high, say 1000. Since the real variables cost nothing, the simplex algorithm will attempt to drive the artificial variables (which initially constitute a basic feasible solution, obviously) out of the basis. Once they have been driven out, the cost function may be replaced with the real one, and with the artificial variables now deleted from the problem, the simplex algorithm will continue on to solve the original problem. Pitfalls which may arise with this method will not be discussed here. For a discussion see (1).

A sample linear programming problem follows:

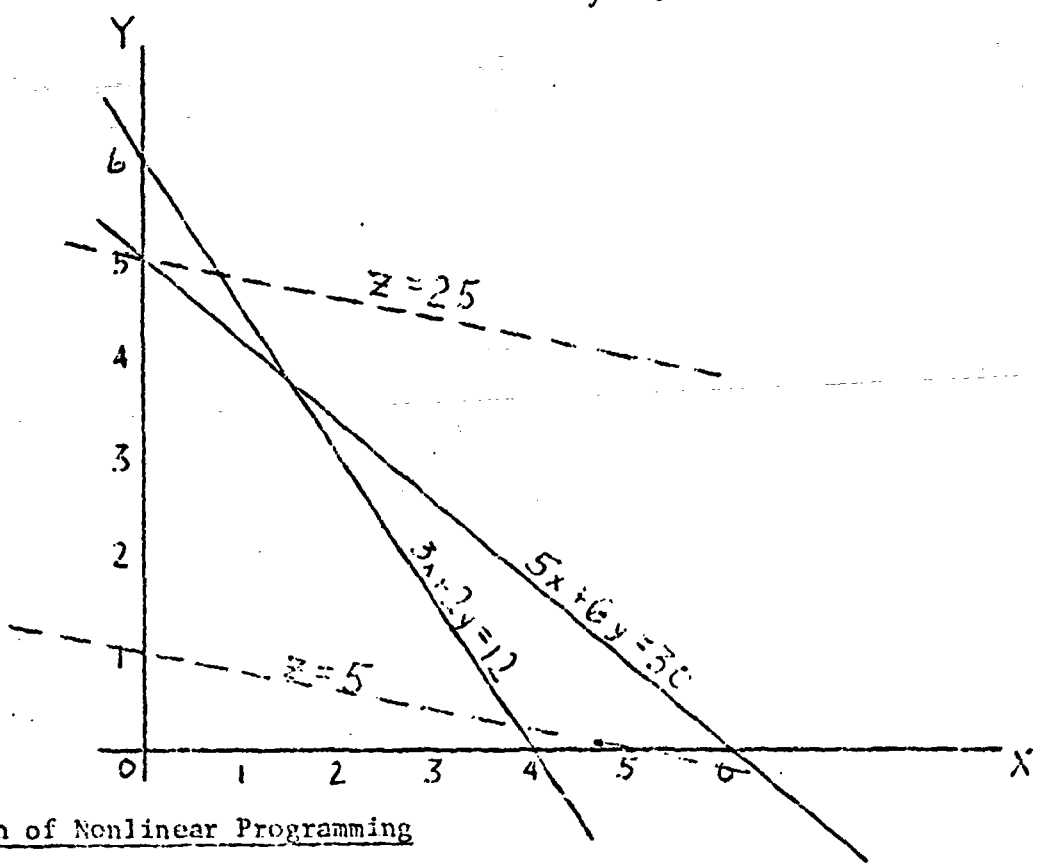
Maximize $Z = x + 5y$

Subject to $5x + 6y \leq 30$

$3x + 2y \leq 12$

$x \geq 0$

$y \geq 0$



Discussion of Nonlinear Programming

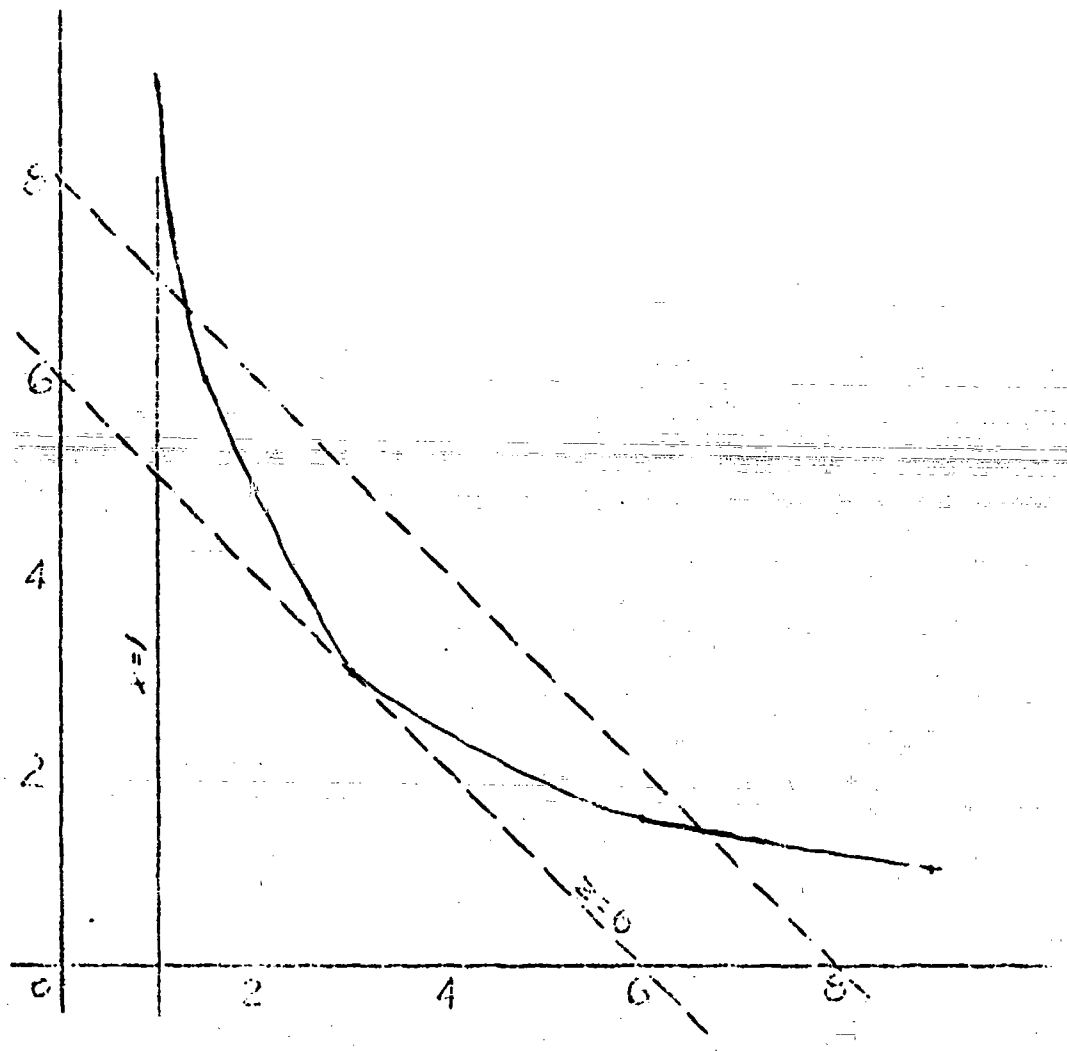
Although on the surface a nonlinear programming problem would appear to be much the same as a linear programming problem, there are in reality many differences, each of which add to the complexity of the nonlinear problem.

The first complexity is concerned with the convexity properties of the set of feasible solutions. Since convexity will not be covered

here (see (1)), we will be satisfied to say that, in the linear case, convergence to a unique answer is guaranteed (if there is any optimum answer) by the convexity of the set of feasible solutions. This convexity is guaranteed by the fact that the constraints are linear inequalities and thus define hyperplanes which separate half spaces which are convex sets. Without the proper convexity properties, which is a common problem in nonlinear programming, solutions leading to local optima can occur, and therefore, the solution obtained may not be the global optimal solution. Fortunately, the problems addressed to date in this research do not have this difficulty, but it was thought pertinent at this point to mention that the problem could occur.

The other principal problem with nonlinear programming is that in many cases the optimum solution will occur not at a vertex of the feasible solution set, but along an edge. You will remember that the simplex algorithm depended upon searching vertices. It turns out that the guarantee of convergence in a finite number of iterations is dependent upon this fact, since there are only a finite number of vertices. However, if all possible boundary points are considered as possible solutions, then there are obviously uncountably infinite possible solutions, and a mere search, even though ordered, cannot be guaranteed to converge finitely. The following will illustrate the nonlinear problem:

$$\begin{aligned} &\text{Min} && Z = x + y \\ &\text{Subject to} && xy \geq 9 \\ &&& x \geq 1 \end{aligned}$$



Assuming that one is faced with solving a nonlinear programming problem which has a feasible solution set which is convex, i.e. a unique solution, one method for solving the problem is the method of separable programming. This method requires that each of the constraint functions, and the objective function, may be written as a sum of functions of one variable. Not all problems are of this type, however, by manipulation, most can be reformulated in such a way that they are separable (see (1)).

Given a separable programming problem, it may be reformulated as a linear programming problem in the following manner. For each variable,

that variable axis is partitioned arbitrarily over the range of conceivable values for that variable. If one now defines λ_i such that $X = \sum\lambda_i x_i$, $\sum\lambda_i = 1$ and if more than one λ_j is positive, then there can be at most two and these must be adjacent. The λ_i are interpolative constants, and therefore, if $g_i = g(X_i)$, the separate functions in the constraints may be written as $\sum\lambda_i g_i$. Therefore, we are left with a linear programming problem with the λ_i as variables, subject to the added constraints that the λ_i for each original variable sum to one, and the condition that, for λ corresponding to the same original variable, no more than two λ_i may be positive, and these two must be adjacent. The method of artificial variables may be used to obtain an initial feasible solution, the same as in normal linear programming problems.

Applications to Least Squares Estimation Problems

Given a least squares estimation problem with a mathematical model $AX = LW$ (possibly the result of linearization of a more complicated model), we will now attempt to apply nonlinear programming to the cost optimization of the problem of finding a set of observations which will allow the values of the propagated errors to fall within certain bounds. Before dealing with specific problems, the problem will be discussed in general.

It has been stated previously that the variance-covariance matrix $\sum_X$, for the adjusted coordinates is given by $(A^T \sum_{LL} A)^{-1}$. The matrix A is of course, constant. However, $\sum_{LL}$ is a function of several parameters which influence the variance of each observation. Let the set of parameters which influence the accuracy of the i^{th} observation be denoted by a vector M_i. Then one arrives at a nonlinear programming problem of the form

$$\min C_i(M_i)$$

subject to

$$(A^T \sum_{LL}^{-1}(M_1,\ldots,M_n)A)^{-1} \leq B.$$

Where $C_i(M_i)$ represents the cost of making the i^{th} observation, and $\sum_{LL}^{-1}(M_i,\ldots,M_n)$ represents the inverse of the variance-covariance matrix written as a function of $M_1,\ldots,M_n$. The reader will notice that, in order to arrive at the constraint equations, a matrix of functions must be inverted. As no straightforward method for doing this exists to the knowledge of the author, some sort of approximate method must be employed. One such method, which met with rather limited success, was employed by the author in (1) and in a paper given on this same topic at the 1969 ACSM-ASP Convention in Washington, D.C. (2).

The author's research efforts during the past year have been almost exclusively directed toward finding a better approximation to this inverse, and it is believed that the approximation which will be presented will alleviate all of the shortcomings of the previous method, with the added advantage that it is materially faster in convergence than the older method.

The new method is, strangely enough, based upon a common error committed in the propagation of variance-covariance matrices for least squares estimators involving nonlinear functions. This is to assume that the parameter values obtained by using only a rough estimate of the true variance-covariance matrix of the observations is correct and to propagate using this information. Thus it is assumed that

$$X = (A^T \hat{\Sigma}_{LL}^{-1} A)^{-1} A^T \hat{\Sigma}_{LL}^{-1} L$$

where $\hat{\Sigma}_{LL}$ is the approximation to the variance-covariance matrix. From statistical theory we know that, if PQ = T and the variance-covariance matrix of Q is Σ_{qq} then

$$\Sigma_{TT} = P \Sigma_{qq} P^T.$$

Therefore,

$$\Sigma_{xx} = (A^T \hat{\Sigma}_{LL}^{-1} A)^{-1} A^T \hat{\Sigma}_{LL}^{-1} \Sigma_{LL} \hat{\Sigma}_{LL}^{-1} A (A^T \Sigma_{LL}^{-1} A)^{-1}.$$

Although the above form is theoretically incorrect (it is correct if $\hat{\Sigma}_{LL} = \Sigma_{LL}$) it turns out that, even for rather crude initial approximations $\hat{\Sigma}_{LL}$, that the estimate of Σ_{xx} obtained by the above method is surprisingly good. For a small problem, this will be shown by the following example:

$$A = \begin{bmatrix} 0 & 0 & -1 \\ -178.6 & 103.1 & -1 \\ .5 & .866 & 0 \end{bmatrix}$$

$$\hat{\Sigma}_{LL} = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & .01 \end{bmatrix}$$

$$\Sigma_{LL} = \begin{bmatrix} .75 & .25 & 0 \\ .25 & .75 & 0 \\ 0 & 0 & .0001 \end{bmatrix}$$

for these matrices

$$(A^T\hat{\Sigma}_{LL}^{-1}A)^{-1} A^T\hat{\Sigma}_{LL}^{-1} \Sigma_{LL} \hat{\Sigma}_{LL}^{-1} A(A^T\hat{\Sigma}_{LL}^{-1}A)^{-1} = (A^T\Sigma_{LL}^{-1}A)^{-1}$$

to four significant figures.

Consideration of the above phenomenon brings to mind a possible solution for the method to be used to approximate the inversion of the matrix $(A^T\Sigma_{LL}^{-1}A)$. First one makes a good guess as to what the optimal values of the variables will be (it turns out that convergence may be

achieved for almost any guess, so the actual values of the guess are unimportant). Then one solves the nonlinear programming problem, using

$$(A^T\hat{\Sigma}_{LL}^{-1}A)^{-1}\ A^T\hat{\Sigma}_{LL}^{-1}\Sigma_{LL}\hat{\Sigma}_{LL}^{-1}A\ (A^T\hat{\Sigma}_{LL}^{-1}A)^{-1}$$

as the left hand side of the constraints (with Σ_{LL}, of course, written as a matrix of functions). The solution vector from this iteration is then used to form a new $\hat{\Sigma}_{LL}$ for the next iteration.

In practice it has been found that the algorithm obtains convergence for most problems within four to five iterations. The method has never failed to converge on a problem. A detailed discussion of how the method can actually be used on a rather small survey network will follow in the next section.

Application to Several Small Survey Problems

It was decided to try the method on several small horizontal control survey networks combining both direction and distance measurements. There is nothing particularly special about this problem; it simply presents a problem of small enough magnitude to be handled without sophisticated programming methods, and yet one with enough complexity so as to pose some challenge to the method.

Using regression analysis techniques, a model for the variance-covariance matrix for a station adjusted set of directions as a function of the

number of pointings made on each point was derived. The variance of the average of n readings on a distance is σ^2/n, where σ^2 is the variance of one reading, so no approximate mathematical model was needed for distances.

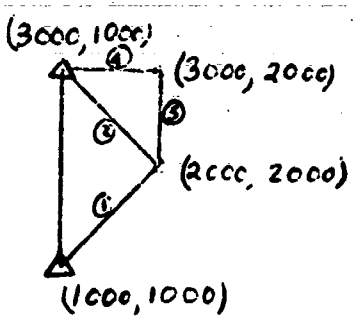
Using these models, one begins by giving the nonlinear programming program guesses at the number of times each measurement is made, a set of required variances for chosen adjusted quantities, a cost vector telling the cost of each repetition of each separate type of measurement, and an observation, or partial derivative, matrix for the least squares adjustment (referred to above as A).

The program then returns the number of times each observation should be made, and the true variance-covariance matrix if the observations are made that number of times.

At present there is no provision for using a nonlinear cost function in the program, although such provision could easily be made. For purposes of testing the method, such a cost function was deemed unnecessary.

Four problems were run, and each will be discussed in detail below:

PROBLEM 1



$$A = \begin{bmatrix} .71 & .71 & 0 & 0 \\ -.71 & .71 & 0 & 0 \\ -1. & 0. & 1 & 0 \\ 0. & 0. & 0 & 1 \end{bmatrix}$$

The coordinates are given in terms of a north, east coordinate system in meters. Distance observations numbered 1, 2, 3, and 4 were made. On all runs it was required that the variances of the north and east coordinates for both unknown points (the points on the right) be under .5 meters. It was assumed that the variance of a single distance observation was one meter^2.

With a cost vector C, C_1 = cost of each individual observation on distance 1, etc., equal to (1,1,1,1) it was found that to meet accuracy requirements, distance 1 must be measured 4 times, two 5 times, three 5 times and four 4 times, with total cost 18.

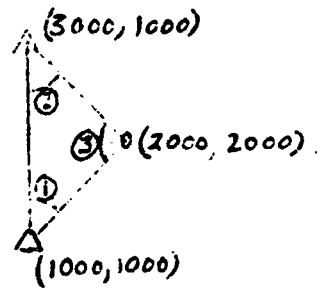
With C = (1,1,10,10), i.e., distances 3 and 4 being 10 times more costly to observe than 1 and 2, we find the number of necessary observations to be 5,5,5,4 with total cost 100.

With C = 1,1,10,1 we get 5,5,5,4 with total cost 64 and with C = 10,10,1,1 we get 5,3,15,4 with total cost 99.

The above numbers are, in all cases, rounded to the next highest even observation, and therefore, some minor changes do not appear. It is, however, evident that even for this very simple problem, the optimum would not be achieved by measuring all distances the same number of times. It should be noted that, since distance four is the only observation determining the east coordinate of the upper unknown point, that

it must be made the same number of times no matter what its cost. This turns out to be true in the results. Also, it should be noted that as the cost of the first two observations increase, the number of times they are to be measured decreases. Why this does not apply to the third observation to the same extent is not known; however, small changes which are masked by the rounding up of the answers did appear.

Problem 2 was a very simple triangulation problem. It consisted of two known stations and one unknown station, with three angles observed,

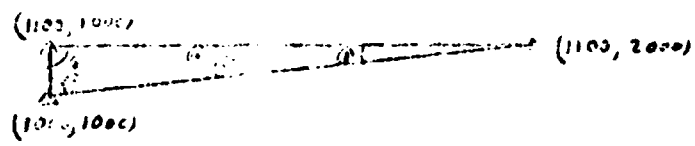


$$A = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ -.71\times10^{-3} & .71\times10^{-3} & -1 & 0 & 0 \\ -.71\times10^{-3} & -.71\times10^{-3} & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ -.71\times10^{-3} & .71\times10^{-3} & 0 & 0 & -1 \\ -.71\times10^{-3} & -.71\times10^{-3} & 0 & 0 & -1 \end{bmatrix}$$

with the last three columns coming from the unknown orientation of the sets of directions. With the requirement that the variances of the coordinates of the unknown point be less than $.5 \times 10^{-4}$ meters^2 and C = 1,10,5 with σ^2 for a single observation equal to 30 secs^2 we get the number of observations to be 50,16,15 respectively with total cost 285. With C = 1,1,10 we get 36,25,1, total cost 71. With C = 1,100,10 we get 50,15,25, total cost 1800. With C = 1,1,1 we get 25,37,3.5,

total cost 65.3. The effect of changing the relative costs upon the optimum solution is easily seen here. It must be remembered that all of the above combinations of measurements meet the accuracy requirements, but that the only differences were in the relative costs associated with each measurement, It is easily seen that measurement costs can greatly influence the configuration of the optimum solution.

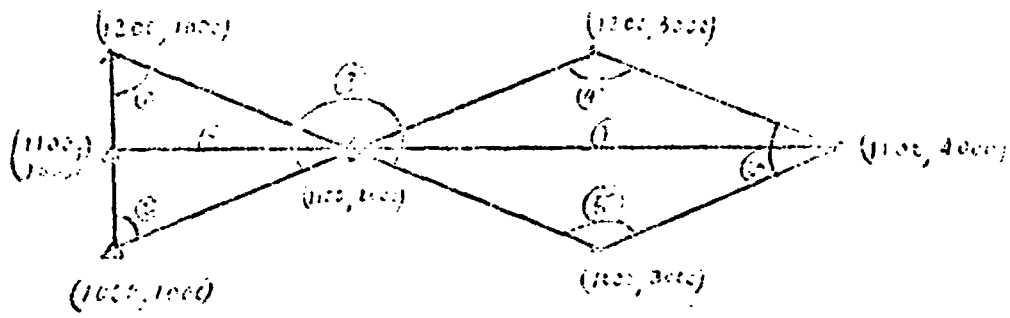
Problem 3 combined direction and distance observations.



The variance of a single direction observation was 3 sec^2 and the variance of a single distance observation was 1 $meter^2$. The variances on north and east of the unknown point .1 x 10^{-2}.

With C = 1,1,1,1,1 the optimum solution was 15,1,1,1,1, total cost 19. With C = 10,5,5,1,1 the solution was 15,1,21,1,1, total cost 167. With C = 200,100,1001,1 the solution was 14,1,1,25,25, total cost 3050. It is obvious from these results that the directions at the unknown point are the most critical observations. Even when they cost 200 times as much as the distances, they must still be made 14 times for an optimal solution. Also, it should be noted that, especially for this problem, the direction observations lend much more strength to the network than the distance observations.

Problem 4 was a somewhat more complex combined angle and distance measurement problem.



The only condition enforced was that the variances of the north and east coordinates of the far right hand point be less than .001 meters^2, The variance for a single direction was taken as 3 seconds ^{2}. With C = 1,1,1,1,1,1,1,1 we get 5,5,5,1,1,1,5,17, total cost 40. With C = 1,1,1,1,1,1,100,100 we get 5,5,50,1,1,1,1,13, total cost 1464. With C = 1,1,4,1,1,1,100,100 we get 15,15,15,1,1,1,1,14, total cost 1593. The influence of the cost vector upon the optimal solution is obvious in this case.

The reader will notice that the total cost is increased rather significantly by the addition of cost constraints. This is logical, since in most cases, the cost of the most critical measurement was increased, and, since this measurement was required essentially without respect to cost, the total cost increases.

It should be mentioned here that we did not allow the values of the variables to go to zero, since we incorporated no provision for handling zero weights (or infinite variances). Also, since our polynominal

approximation ended at 50, no value could exceed 50. In only one case does the restriction of 50 enter in; however, the restriction that the variables could not be less than 1 entered into several problems. A possible solution to this is discussed in the conclusions.

As a comparison, for problem 4 the accuracy specified was about one part in 100,000. If we assumed C = 1,1,1,1,1,1,1,1 and used a set of specifications telling us to measure each direction 16 times (8 sets of both direct and reverse) and to measure each distance 4 times, the total cost would be 104, vs a total cost of 40 for the optimal solution, but worse than that, the results of the survey would not meet required accuracy. This example should make the advantages of this method obvious to the reader.

Discussion of Applications to More General Problems

The method can be extended rather simply to general problems. For problems involving trade offs between various possible methods, one could simply set up the problem as if all observations possible were being made and, if the variance functions are picked so that if an observation is not made, it will have zero weight in the adjustment, then a least cost solution would allow some measurements not to be made at all. Thus the method could be used as a decision tool as to which measurements should be made, or where control points should be placed, etc. Ideally, in this mode, integer programming techniques should be used, so that a binary (0,1) possible range of answers would be possible. This would complicate the programming solution somewhat, but it is certainly feasible.

Another possibility is that the method could be applied separately to each of two or more competitive systems to find out the minimum cost of attaining a given accuracy with each. It would then be simple to see which system would be least costly for accomplishing a given purpose. This application could save considerable expense by eliminating the complete development of two computing systems when one is found to be considerably less cost effective than the other.

It should be realized at this point that the computer programs used to apply this method to these larger problems will be rather large and involved. Therefore, it is not envisioned at this time that this method will be used to obtain an optimum design for each individual project, but only to arrive at an optimum design for a typical project of a class. The other projects in this class could be accomplished in a near optimum manner using the experience gained from the test project.

Conclusion

As can be seen from the sample problem in the last section, the method works relatively well for small problems. The difficulties encountered in completely deleting measurements are being eliminated by using integer programming techniques.

The method has been shown to be feasible for ground survey problems. The author believes it to be feasible for larger and more general classes of problems both in the mapping field, and in other areas of interest. Of course, the actual application of the method may have to be

tailored to the individual problem; however, the general concepts will apply.

It is hoped that, in the next year, larger problems can be attacked and the feasibility of the method for totally different problems can be studied. The author would like to apply the method to such things as optimal location of control for photogrammetric block adjustments, configuration of navigational satellite systems, and other problems.

If the author can be of any assistance to anyone else interested in working in this area, he would be more than willing to discuss the matter further.

BIBLIOGRAPHY

1. Greve, Clifford W., "Design of Survey Networks Using Nonlinear Programming." Ph.D. Thesis, Cornell University, Ithaca, New York, 1969. (Available from Director, U. S. Geological Survey.)

2. Greve, Clifford W., "Optimal Design of Survey Networks." Paper presented to ACSM-ASP Convention. Washington, D. C., March, 1969.

3. Hadley, G., Nonlinear and Dynamic Programming. Addison Wesley Publishing Company, Inc., Reading, Mass., 1964.

ON THE RATE OF CHANGE IN THE SOLUTION SET OF A PERTURBED LINEAR PROGRAM

Stephen M. Robinson
Mathematics Research Center
University of Wisconsin
Madison, Wisconsin

Abstract. In this paper we find bounds for the displacement in the solution set of a system of linear inequalities caused by perturbations in the coefficient matrix and/or the right-hand side, and apply these to estimate the error in the optimal solution of a perturbed linear program. The main result is that if a superconsistent linear program is subjected to a sequence of perturbations approaching zero, for which a uniformly bounded sequence of (primal) solutions to the perturbed program exists, then the distances from those solutions to the solution set of the unperturbed program are of the large order of the perturbations.

1. Introduction.

It is known that under certain regularity conditions the solution set of a linear program (and of much more general programs) is upper semicontinuous under perturbations in the constraints and/or the objective function ([1], [2], [3]). Here "upper semicontinuous" is to be interpreted in the sense applicable to set-valued mappings [1]. However, estimates for the changes in the solution set under such perturbations seem not to have been given. In this paper we develop such estimates. We proceed by first developing bounds for the displacement in the solution set of a system of perturbed linear inequalities, then apply these bounds to the case of a linear program to obtain a bound for the distance from a point solving the perturbed program to the solution set of the unperturbed program. Since this bound is expressed in terms of constants which are shown to exist but which may not be readily computable, it is not likely to provide computable error estimates. However, it does permit us to draw conclusions about the rate at which the solutions approach those of the unperturbed program. These conclusions are summarized in Theorem 7, where we show that if the unperturbed program is superconsistent and if there is a sequence of primal solutions to the perturbed programs which can be uniformly bounded as the perturbations approach zero, then the distances

---

Sponsored by the United States Army under Contract No.: DA-31-124-ARO-D-462. An earlier version of this paper was presented to the 1971 Army Numerical Analysis Conference on 22 April 1971.

The remainder of this paper was reproduced photographically from the author's manuscript.

from those solutions to the solution set of the unperturbed program are of the large order of the perturbations in the coefficient matrix, right-hand side, and objective function.

In the course of obtaining these results, we develop some properties of linear inequalities which may be of independent interest; in particular we show that each real matrix defines a number which behaves, with respect to systems of inequalities involving that matrix, much as the norm of the inverse of a non-singular square matrix does with respect to systems of equations.

We shall use upper and lower case Roman letters for matrices and vectors respectively, and lower case Greek letters for scalars. $\mathbb{R}^k$ denotes k-dimensional real linear space; $\| \ \|_\infty$ and $\| \ \|_1$ are used for the ℓ_∞ and ℓ_1 norms, respectively (see, e.g., [4]). The symbol e is used for a vector with 1 in every component. The transpose of x is written x^T. Subscripts on vectors and matrices refer to components, except at the end of the paper where they indicate membership in a sequence; there should be no confusion as to which is intended in any particular place.

2. Linear Inequalities.

In this section we present various results about linear inequalities, some of which we shall have occasion to use later on. We first show that any real matrix defines a number with the properties mentioned in Section 1. The term "polyhedral norm" means any norm whose unit sphere is a polyhedron.

LEMMA: Let A be an $m \times n$ real matrix and let $\| \ \|_m$ and $\| \ \|_n$ be polyhedral norms on $\mathbb{R}^m$ and $\mathbb{R}^n$ respectively. Then there is a scalar $\mu(A)$, depending only on A, such that for each $b \in \mathbb{R}^m$, either

a. $Ax \leqq b$ has no solution, or

b. $Ax \leqq b$ has a solution $\bar{x}$ with $\|\bar{x}\|_n \leqq \mu(A) \|b\|_m$.

Further, there exists a $b \in \mathbb{R}^m$ such that $Ax \leqq b$ is consistent and for every solution $\bar{x}$ we have $\|\bar{x}\|_n \geqq \mu(A) \|b\|_m$.

PROOF: Let P_m and P_n be the unit polyhedra of $\| \ \|_m$ and $\| \ \|_n$ respectively. Consider the set $B := \{b \mid b \in \mathbb{R}^m$ and $Ax \leqq b$ is solvable$\}$. It is easy to see that $B = A(\mathbb{R}^n) + \mathbb{R}^m_+ = A(\mathbb{R}^n_+) + (-A)(\mathbb{R}^n_+) + \mathbb{R}^m_+$, where $\mathbb{R}^k_+$ is the non-negative orthant of $\mathbb{R}^k$. Thus B, being the sum of three polyhedral convex cones, is itself a polyhedral convex cone. The intersection of B with the unit polyhedron $P_m := \{b \mid b \in \mathbb{R}^m$ and $\|b\|_m \leq 1\}$ is nonempty since $0 \in P_m \cap B$; thus $P_m \cap B$ is a convex polyhedron, and so is the convex hull of a finite number of extreme points, say $\{p_1, \ldots, p_r\}$. Now consider the function $\eta(q)$ defined on B by $\eta(q) := \min\{\eta \mid \eta \geqq 0$ and for some x, $Ax \leqq q$ and $x \in \eta P_n\}$. This function certainly exists for any $q \in B$, since if (x', η') is any feasible pair then $\eta(q)$ can be found by minimizing η over the compact set $\{(x, \eta) \mid Ax \leqq q, \ x \in \eta P_n\}$ $\cap \{\eta' P_n \times [0, \eta']\}$. In addition, $\eta(q)$ is convex on B. To see this, let $q_1, q_2 \in B$ and suppose $\eta(q_1) = \eta_1$, $\eta(q_2) = \eta_2$. Let $Az_1 \leqq q_1$ with $\|z_1\|_n = \eta_1$, and $Az_2 \leqq q_2$ with $\|z_2\|_n = \eta_2$. Now choose any

$\lambda \in [0, 1]$ and consider the point $q_3 := \lambda q_1 + (1-\lambda)q_2$. Let $z_3 := \lambda z_1 + (1-\lambda)z_2$; then clearly $Az_3 = \lambda Az_1 + (1 - \lambda)Az_2 \leq \lambda q_1 + (1 - \lambda)q_2 = q_3$, and $\|z_3\|_n = \|\lambda z_1 + (1 - \lambda)z_2\|_n \leq \lambda \|z_1\|_n + (1 - \lambda)\|z_2\|_n = \lambda\eta_1 + (1-\lambda)\eta_2$. Thus $\eta(q_3) \leq \|z_3\|_n \leq \lambda\eta(q_1) + (1 - \lambda)\eta(q_2)$, which proves the convexity of η. Consider the restriction of η to $P_m \cap B$; since $P_m \cap B$ is the convex hull of $p_1, \ldots, p_r$ and since a convex function on a convex polyhedron attains its maximum at one of the extreme points (because any point in the polyhedron is a convex combination of the extreme points), it follows that for each $q \in P_m \cap B$ we have $\eta(q) \leq \eta(p_M) :=$ $\max\{\eta(p_i) \mid 1 \leq i \leq r\} =: \mu$.

Now let $b \in \mathbb{R}^m$. If $Ax \leq b$ has no solution then conclusion (a.) holds; thus, let $Ax \leq b$ be solvable. If $b = 0$ then $\bar{x} = 0$ is a solution and $\|\bar{x}\|_n \leq \mu \|b\|_m$. If $b \neq 0$, let $\hat{b} := b/\|b\|_m$; then $\hat{b} \in P_m \cap B$, so there is an $\hat{x}$ such that $A\hat{x} \leq \hat{b}$ and $\|\hat{x}\|_n \leq \mu$. Multiplying by $\|b\|_m$, we find that with $\bar{x} := \|b\|_m \hat{x}$ we have $A\bar{x} \leq b$ and $\|\bar{x}\|_n \leq \mu \|b\|_m$. Thus conclusion (b.) holds if conclusion (a.) does not.

Finally, consider p_M. If $p_M = 0$ we have $\mu = 0$, and clearly any solution of $Ax \leq 0$ has $\|x\|_n \geq 0 = \mu \|p_M\|_m$. Assume $p_M \neq 0$ and $0 < \|p_M\|_m < 1$. Let $\lambda := 1/(1 + \|p_M\|_m)$; then $0 < \lambda < 1$. Both $r_1 := \|p_M\|_m p_M$ and $r_2 := \|p_M\|_m^{-1} p_M$ lie in B (because it is a cone) and in P_m (because $\|r_1\|_m < 1 = \|r_2\|_m$), and it is easily seen that $p_M = \lambda r_1 + (1-\lambda)r_2$, which contradicts the fact that p_M is an extreme point of $P_m \cap B$. Thus $\|p_M\|_m = 1$. We know that for each $\bar{x}$ with $A\bar{x} \leq p_M$ we have $\|\bar{x}\|_n \geq \mu$, but since $\|p_M\|_m = 1$ we have $\|\bar{x}\|_n \geq$ $\geq \mu \|p_M\|_m$. This completes the proof.

We remark that in the case of non-polyhedral norms, the first conclusion of the lemma remains valid since any two norms on a finite-dimensional space are equivalent.

For the remainder of this paper we shall let $\| \ \|_m$ and $\| \ \|_n$ be the ℓ_∞ norm on the spaces $\mathbb{R}^m$ and $\mathbb{R}^n$ respectively.

To indicate one application of this lemma, we mention that it may be used to obtain bounds for the dual variables of linear programs in terms of the primal variables and the objective function, as shown in the following theorem.

THEOREM 1: Let A be an $m \times n$ matrix and let $b \in \mathbb{R}^m$. Suppose $\{x \mid Ax \leq b\} \neq \emptyset$. Then there is a constant ρ, depending only on A and b, such that for every $p \in \mathbb{R}^n$ and every optimal solution $\bar{x}$ (if any exists) of the program $\min\{p^T x \mid Ax \leq b\}$, there is a corresponding dual solution $\bar{u} \in \mathbb{R}^m_+$ with

$$\|\bar{u}\|_\infty \leq \rho \|p\|_\infty \max(1, \ \|\bar{x}\|_1) \ .$$

PROOF: The vector $\bar{u}$ is a dual solution of the cited program if and only if it satisfies

$$\bar{u}^T[A \ \ b] = -p^T[I \ \ \bar{x}]$$

and

$$\bar{u} \geq 0 \ .$$

By hypothesis, $\bar{x}$ is a primal optimal solution, so at least one such $\bar{u}$ exists. Rewriting this system as

$$[A \quad b \quad -A \quad -b \quad -I]^T \bar{u} \leq [-p^T[I \quad \bar{x}] \quad p^T[I \quad \bar{x}] \; 0]^T ,$$

letting ρ equal the number μ associated by the preceding lemma with the matrix on the left-hand side and estimating the ℓ_∞ norm of the right-hand side by $\|p\|_\infty \max(1, \|\bar{x}\|_1)$, we find that a dual solution $\bar{u}$ will exist with $\|\bar{u}\|_\infty \leq \rho \|p\|_\infty \max(1, \|\bar{x}\|_1)$. This completes the proof.

Unfortunately, the above result is useful only in case A and b remain constant. Since we are interested in the case in which all three of A, b and p may vary, we shall want to find out something about the behavior of $\mu(A)$ as A varies; in particular we should like to know whether (and when) $\mu(A)$ is continuous in A.

It is immediately clear that $\mu(A)$ is not always continuous; for example, consider the set of 1×1 real matrices. We have $\mu([0]) = 0$, but for $\alpha \neq 0$ $\mu([\alpha]) = |\alpha|^{-1}$, so μ is not continuous at 0. This situation is analogous to the problem of singularity for square matrices; to resolve it we introduce a requirement analogous to that of full rank.

DEFINITION: We say that a matrix A satisfies the PLI condition if and only if the rows of A are positively linearly independent; that is, $u^T A = 0$ and $u \geq 0$ implies $u = 0$.

THEOREM 2: Let A be an $m \times n$ matrix. The inequalities $Ax \leq b$ are solvable for every right-hand side $b \in \mathbf{R}^m$ if and only if A satisfies the PLI condition.

PROOF: Follows directly from Gordan's theorem [5].

If A satisfies the PLI condition, then not only is $\mu(A)$ continuous at A but we can even find estimates for its variation near A. These are given in the following analogue of Banach's lemma for nonsingular linear transformations.

THEOREM 3: Let A be an $m \times n$ matrix satisfying the PLI condition. Then for every $m \times n$ matrix H with $\mu(A)\|H\|_\infty < 1$, we have:

a. $A + H$ satisfies the PLI condition, and

b. $\mu(A)(1 + \mu(A)\|H\|_\infty)^{-1} \leq \mu(A + H) \leq \mu(A)(1 - \mu(A)\|H\|_\infty)^{-1}$.

PROOF: By Theorem 2 and the definition of $\mu(A)$, we can find an $\bar{x}$ such that $A\bar{x} \leq -e$ and $\|\bar{x}\|_\infty \leq \mu(A)$. Now assume that $\mu(A)\|H\|_\infty < 1$; then $(A + H)\bar{x} = A\bar{x} + H\bar{x} \leq -e + \|H\|_\infty \|\bar{x}\|_\infty e \leq -(1 - \mu(A)\|H\|_\infty)e < 0$, so by Gordan's theorem [5] the matrix $A + H$ satisfies the PLI condition.

#1134

Now let $b \in \mathbb{R}^m$ be arbitrary. As above, there is some x such that $Ax \leqq b$ and $\|x\|_\infty \leqq \mu(A)\|b\|_\infty$. Then

$$(A + H)[x + (1 - \mu(A)\|H\|_\infty)^{-1}\mu(A)\|H\|_\infty\|b\|_\infty\bar{x}] \leqq b + Hx +$$

$$+ (1 - \mu(A)\|H\|_\infty)^{-1}\mu(A)\|H\|_\infty\|b\|_\infty[-(1 - \mu(A)\|H\|_\infty)e] \leqq$$

$$\leqq b + \|H\|_\infty\mu(A)\|b\|_\infty - \mu(A)\|H\|_\infty\|b\|_\infty = b .$$

Further,

$$\|x + (1 - \mu(A)\|H\|_\infty)^{-1}\mu(A)\|H\|_\infty\|b\|_\infty\bar{x}\|_\infty \leqq \mu(A)\|b\|_\infty +$$

$$+ (1 - \mu(A)\|H\|_\infty)^{-1}\mu(A)^2\|H\|_\infty\|b\|_\infty = \mu(A)(1 - \mu(A)\|H\|_\infty)^{-1}\|b\|_\infty ,$$

so since b was arbitrary we must have $\mu(A + H) \leqq \mu(A)(1 - \mu(A)\|H\|_\infty)^{-1}$.

Again let $b \in \mathbb{R}^m$ be arbitrary. Choose y so that $(A + H)y \leqq b$ with $\|y\|_\infty \leqq \mu(A + H)\|b\|_\infty$, and consider the vector $y + \mu(A + H)\|H\|_\infty\|b\|_\infty\bar{x}$, in which $\bar{x}$ is as previously defined. Then we have

$$A(y + \mu(A + H)\|H\|_\infty\|b\|_\infty\bar{x})$$

$$\leqq (A + H)y - Hy - \mu(A + H)\|H\|_\infty\|b\|_\infty e$$

$$\leqq (A + H)y \leqq b,$$

and

$$\|y + \mu(A + H)\|H\|_\infty\|b\|_\infty\bar{x}\|_\infty \leqq \mu(A + H)(1 + \mu(A)\|H\|_\infty)\|b\|_\infty ,$$

so $\mu(A) \leqq \mu(A + H)(1 + \mu(A)\|H\|_\infty)$, or $\mu(A)(1 + \mu(A)\|H\|_\infty)^{-1} \leqq \mu(A + H)$. This completes the proof.

#1134

Consideration of the 1×1 matrix $A = [1]$ and the class of matrices $H = [\epsilon]$, $|\epsilon| < 1$, shows that the above bounds are sharp.

We now turn our attention from the study of $\mu(A)$ to consideration of perturbed systems of linear inequalities. The theorem we obtain here will be our main tool in studying the behavior of perturbed linear programs.

Consider a system of linear inequalities $Ax \leq b$, and assume this system has at least one solution. We shall consider another solvable system of the same dimensions, $\hat{A}x \leq \hat{b}$, and pose the following problem: Given a solution $\hat{x}$ to the second system, find a "small" ℓ_∞ sphere about $\hat{x}$ such that a solution of the first system must lie within that sphere. Finding the smallest such sphere is equivalent to determining the ℓ_∞ distance from $\hat{x}$ to the solution set of the first system. This problem is solved by the following theorem.

THEOREM 4: Let A and $\hat{A}$ be $m \times n$ real matrices, and let b and $\hat{b}$ be vectors in $\mathbb{R}^m$. Suppose $\bar{x}$ is such that $A\bar{x} \leq b$. Then there exist constants μ and ν depending only on A, b and $\bar{x}$, such that for each $\hat{x}$ with $\hat{A}\hat{x} \leq \hat{b}$ there exists an $\tilde{x}$ with $A\tilde{x} \leq b$ and

$$\|\tilde{x} - \hat{x}\|_\infty \leq (\mu + \nu \|\hat{x} - \bar{x}\|_\infty)(\|A - \hat{A}\|_\infty \|\hat{x}\|_\infty + \|b - \hat{b}\|_\infty) .$$

PROOF: Since $\bar{x}$ is a feasible point (solution) for the inequalities $Ax \leq b$, it follows that $b - A\bar{x} \geq 0$. Define $I_1 := \{i \mid (b - A\bar{x})_i > 0\}$ (sometimes called the carrier of $b - A\bar{x}$) and $I_2 := \{1, \ldots, m\} \setminus I_1 = \{i \mid (b - A\bar{x})_i = 0\}$. Rearrange the rows of A, $\hat{A}$, b and $\hat{b}$, if necessary,

so that all the members of I_1 precede all the members of I_2. Thus we obtain a partitioning of A and b (as well as of $\hat{A}$ and $\hat{b}$) as follows:

$$A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}, \qquad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix},$$

where $b_1 > A_1\bar{x}$ and $b_2 = A_2\bar{x}$. Of course, either I_1 or I_2 could be empty, in which case no partitioning would be necessary.

Now consider the linear program:

$$\left.\begin{array}{l} \text{Find } \bar{y} \text{ and } \bar{\epsilon} \text{ to minimize } \epsilon \\ \text{subject to } -\epsilon e \leq (\bar{x} + y) - \hat{x} \leq \epsilon e \\ A(\bar{x} + y) \leq b \,. \end{array}\right\} \qquad (1)$$

It is clear that y and some ϵ are feasible for (1) if and only if $\bar{x} + y$ satisfies $A(\bar{x} + y) \leq b$. Consequently, if we can solve (1), the value of $\bar{\epsilon}$ will yield the minimum of $\|x - \hat{x}\|_\infty$ over all x satisfying $Ax \leq b$.

Using the partitioning outlined above, the dual of (1) can be written as:

$$\left.\begin{array}{l} \text{Find } \bar{u}_1, \bar{u}_2, \bar{v}_1, \bar{v}_2 \\ \quad \text{to maximize } (u_1 - u_2)^T(\hat{x}-\bar{x}) - v_1^T(b_1 - A_1\bar{x}) \\ \text{subject to } (u_1 - u_2)^T - v_1^TA_1 - v_2^TA_2 = 0 \\ \qquad (u_1 + u_2)^Te = 1 \\ \qquad u_1, u_2, v_1, v_2 \geq 0 \,. \end{array}\right\} \qquad (2)$$

Both (1) and (2) are obviously feasible; hence by the duality theorem of linear programming [5] they are both solvable and the extrema are equal. Setting $\tilde{x} := \bar{x} + \bar{y}$, we then see that $A\tilde{x} \leq b$ and that for any $\tilde{v}_2$ such that $\bar{u}_1$, $\bar{u}_2$, $\bar{v}_1$, and $\tilde{v}_2$ satisfy the constraints of (2), we have

$$0 \leq \|\tilde{x} - \hat{x}\|_\infty = (\bar{u}_1 - \bar{u}_2)^T(\hat{x} - \bar{x}) - \bar{v}_1^T(b_1 - A_1\bar{x})$$

$$= (\bar{u}_1 - \bar{u}_2 - \bar{v}_1^T A_1 - \tilde{v}_2^T A_2)(\hat{x} - \bar{x})$$

$$+ \bar{v}_1^T(A_1\hat{x} - b_1) + \tilde{v}_2^T(A_2\hat{x} - b_2)$$

$$= [\bar{v}_1^T \tilde{v}_2^T](A\hat{x} - b)$$

$$= [\bar{v}_1^T \tilde{v}_2^T]\{(\hat{A}\hat{x} - \hat{b}) + (A - \hat{A})\hat{x} - (b - \hat{b})\}$$

$$\leq [\bar{v}_1^T \tilde{v}_2^T][(A - \hat{A})\hat{x} - (b - \hat{b})] ,$$

where we have used the constraints of (2) and the facts that $\hat{A}\hat{x} \leq \hat{b}$ and $A_2\bar{x} = b_2$.

Let $\delta^{-1} := \min\{(b - A\bar{x})_i \mid i \in I_1\} > 0$. Since $\|\tilde{x} - \hat{x}\|_\infty \geq 0$, we have $(\bar{u}_1 - \bar{u}_2)^T(\hat{x} - \bar{x}) \geq \bar{v}_1^T(b_1 - A_1\bar{x}) \geq \delta^{-1}\bar{v}_1^T e$, and thus $\|\bar{v}_1\|_1 = \bar{v}_1^T e \leq \delta(\bar{u}_1 - \bar{u}_2)^T(\hat{x} - \bar{x}) \leq \delta\|\hat{x} - \bar{x}\|_\infty(\|\bar{u}_1\|_1 + \|\bar{u}_2\|_1) = \delta\|\hat{x} - \bar{x}\|_\infty(\bar{u}_1 + \bar{u}_2)^T e =$ $= \delta\|\hat{x} - \bar{x}\|_\infty$. The only requirement that we placed on $\tilde{v}_2$ was that it satisfied $A_2^T\tilde{v}_2 = (u_1 - u_2) - A_1^T\bar{v}_1$; by the lemma there exist a constant μ, depending only on A_2, and a solution $\bar{v}_2$ with $\|\bar{v}_2\|_1 \leq \mu\|\bar{u}_1 - \bar{u}_2 - A_1^T\bar{v}_1\|_1 \leq$ $\leq \mu(\|\bar{u}_1\|_1 + \|\bar{u}_2\|_1 + \|A_1\|_\infty\|\bar{v}_1\|_1) \leq \mu(1 + \delta\|A_1\|_\infty\|\hat{x} - \bar{x}\|_\infty)$. Thus, setting $\tilde{v}_2 = \bar{v}_2$, we obtain

$$\|\tilde{x} - \hat{x}\|_\infty \leq \|[\bar{v}_1^T \; \bar{v}_2^T]\|_\infty (\|A - \hat{A}\|_\infty \|\hat{x}\|_\infty + \|b - \hat{b}\|_\infty)$$

$$= (\|\bar{v}_1\|_1 + \|\bar{v}_2\|_1)(\|A - \hat{A}\|_\infty \|\hat{x}\|_\infty + \|b - \hat{b}\|_\infty)$$

$$\leq [\mu + (1 + \mu \|A_1\|_\infty)\delta \|\hat{x} - \bar{x}\|_\infty](\|A - \hat{A}\|_\infty \|\hat{x}\|_\infty + \|b - \hat{b}\|_\infty) ,$$

which, with $\nu := (1 + \mu \|A_1\|_\infty)\delta$, completes the proof.

We remark that if $b \geq 0$, then $\bar{x} = 0$ is a choice that satisfies the requirements of the theorem. Also, in case $Ax \leq b$ is superconsistent and $A\bar{x} < b$, then $I_2 = \emptyset$ and the above estimate takes the simpler form

$$\|\tilde{x} - \hat{x}\|_\infty \leq \delta \|\hat{x} - \bar{x}\|_\infty (\|A - \hat{A}\|_\infty \|\hat{x}\|_\infty + \|b - \hat{b}\|_\infty) ,$$

since the lemma is not required.

3. Linear Programs.

We now apply Theorem 4 to the problem of bounding the error in a perturbed linear program. The result we obtain in this case is stated in the following theorem.

THEOREM 5: Consider the linear programs

$$\min_x \{p^T x \mid Ax \leq b\} \tag{3}$$

and

$$\min_x \{\hat{p}^T x \mid \hat{A}x \leq \hat{b}\} , \tag{4}$$

where A and $\hat{A}$ are $m \times n$ real matrices, and where b, $\hat{b}$ and p, $\hat{p}$ are vectors in $\mathbb{R}^m$ and $\mathbb{R}^n$ respectively. Let $\bar{x}$ solve (3). Then there exist constants θ, ζ, and σ depending only on A, b, p and $\bar{x}$, such that if $\hat{x}$ and $\hat{u}$ are respectively primal and dual solutions of (4), and if we define

$$\eta := \zeta + \theta \|\bar{x} - \hat{x}\|_\infty ,$$

$$\xi := \zeta + \sigma \|\bar{x} - \hat{x}\|_\infty ,$$

$$\alpha := \eta \|\hat{x}\|_\infty + \xi \|\hat{u}\|_1 \|\bar{x}\|_\infty ,$$

$$\beta := \eta + \xi \|\hat{u}\|_1 ,$$

and

$$\gamma := \eta \|\hat{x}\|_\infty + \xi \|\bar{x}\|_\infty ,$$

then there is a vector $\tilde{x} \in \mathbb{R}^n$ such that $\tilde{x}$ solves (3) and

$$\|\tilde{x} - \hat{x}\|_\infty \leq \alpha \|A - \hat{A}\|_\infty + \beta \|b - \hat{b}\|_\infty + \gamma \|p - \hat{p}\|_1 .$$

PROOF: Suppose $p^T\bar{x} = \lambda$ and $\hat{p}^T\hat{x} = \hat{\lambda}$. We shall first obtain a lower bound on $\lambda - \hat{\lambda}$. Since $\hat{u}$ solves the dual of (4),

$$\max_u \{-u^T\hat{b} \mid u^T\hat{A} + \hat{p}^T = 0, \ u \geq 0\}, \tag{5}$$

we have $\hat{u}^T(\hat{A}\hat{x} - \hat{b}) = 0$ and

$$\hat{p}^T\hat{x} = -\hat{u}^T\hat{b}$$

$$= -\hat{u}^T\hat{b} + (\hat{u}^T\hat{A} + \hat{p}^T)\bar{x}$$

$$= \hat{p}^T\bar{x} + \hat{u}^T(\hat{A}\bar{x} - \hat{b}),$$

so

$$\lambda - \hat{\lambda} = p^T\bar{x} - \hat{p}^T\hat{x} = (p - \hat{p})^T\bar{x} - \hat{u}^T(\hat{A}\bar{x} - \hat{b})$$

$$\geqq (p - \hat{p})^T\bar{x} - \hat{u}^T(\hat{A}\bar{x} - \hat{b}) + \hat{u}^T(A\bar{x} - b)$$

$$= (p - \hat{p})^T\bar{x} + \hat{u}^T[(A - \hat{A})\bar{x} - (b - \hat{b})]. \qquad (6)$$

We remark that we could also have obtained an upper bound in terms of $\hat{x}$ and a dual solution to (3). Extensions of this procedure to convex programs are relatively easy to obtain via the Kuhn-Tucker saddlepoint conditions.

Now we consider the two systems of linear inequalities

$$\begin{bmatrix} A \\ p^T \end{bmatrix} x \leqq \begin{bmatrix} b \\ \lambda \end{bmatrix} \qquad (7)$$

and

$$\begin{bmatrix} \hat{A} \\ \hat{p}^T \end{bmatrix} x \leqq \begin{bmatrix} \hat{b} \\ \hat{\lambda} \end{bmatrix}. \qquad (8)$$

Clearly any feasible point of (7) or (8) is an optimal solution of (3) or (4) respectively. We now apply Theorem 4 to obtain the result we are looking for. Just as in the proof of that theorem we find that there is an optimal solution $\tilde{x}$ of (3) satisfying

$$\|\tilde{x} - \hat{x}\|_\infty = (\bar{u}_1 - \bar{u}_2)^T(\hat{x} - \bar{x}) - \bar{v}_1^T(b_1 - A_1\bar{x}), \qquad (9)$$

with optimal dual variables $\bar{u}_1$, $\bar{u}_2$ and $\bar{v}_1$ satisfying the following constraints:

$$\left.\begin{aligned} \bar{u}_1 - \bar{u}_2 - \bar{v}_1^T A_1 - \tilde{v}_2^T A_2 - \tilde{\omega} p^T &= 0 \\ (\bar{u}_1 + \bar{u}_2)^T e &= 1 \\ \bar{u}_1, \bar{u}_2, \bar{v}_1, \tilde{v}_2, \tilde{\omega} &\geqq 0 . \end{aligned}\right\} \qquad (10)$$

Here A_1, A_2, b_1 and b_2 are defined just as in Theorem 4. Note that once $\bar{u}_1$, $\bar{u}_2$ and $\bar{v}_1$ are fixed, the variables $\tilde{v}_2$ and $\tilde{\omega}$ in (10) may assume any non-negative values consistent with the equations of (10); the duality theorem guarantees the existence of at least one non-negative pair satisfying those equations.

By arguing as in the proof of Theorem 4, we obtain the following results, in which δ is defined as previously.

a. $\|\bar{v}_1\|_1 \leq \delta \|\hat{x} - \bar{x}\|_\infty$.

b. There are feasible solutions $\bar{v}_2$ and $\bar{\omega}$ to (10), and a constant ζ depending only on A_2 and p, such that $\|[\bar{v}_2^T \; \bar{\omega}]\|_\infty \leqq \zeta \|\bar{u}_1 - \bar{u}_2 - A_1^T \bar{v}_1\|_1 \leqq$
$\leqq \zeta(1 + \delta \|A_1\|_\infty \|\hat{x} - \bar{x}\|_\infty)$.

c. $\|\tilde{x} - \hat{x}\|_\infty \leqq [\bar{v}_1^T \; \bar{v}_2^T \; \bar{\omega}] \left[\begin{pmatrix} A - \hat{A} \\ p^T - \hat{p}^T \end{pmatrix} \hat{x} - \begin{pmatrix} b - \hat{b} \\ \lambda - \hat{\lambda} \end{pmatrix} \right]$.

Using (a.), (b.), and (c.), we obtain with $\theta := (1 + \zeta \|A_1\|_\infty)\delta$ and $\eta := \zeta + \theta \|\hat{x} - \bar{x}\|_\infty$,

$$\|\tilde{x} - \hat{x}\|_\infty \leq \eta\left(\left\| \begin{matrix} A - \hat{A} \\ p^T - \hat{p}^T \end{matrix} \right\|_\infty \|\hat{x}\|_\infty + \|b - \hat{b}\|_\infty\right) - \bar{\omega}(\lambda - \hat{\lambda}) . \qquad (11)$$

But from (6), we obtain

$$-\bar{\omega}(\lambda - \hat{\lambda}) \leq \bar{\omega}\{\|p - \hat{p}\|_1 \|\bar{x}\|_\infty + \|\hat{u}\|_1(\|A - \hat{A}\|_\infty \|\bar{x}\|_\infty + \|b - \hat{b}\|_\infty)\}, \qquad (12)$$

and we have

$$0 \leq \bar{\omega} \leq \|[\bar{v}_2^T \bar{\omega}]\|_\infty \leq \zeta(1 + \delta \|A_1\|_\infty \|\hat{x} - \bar{x}\|_\infty) =: \zeta + \sigma \|\hat{x} - \bar{x}\|_\infty . \qquad (13)$$

Putting together (11), (12) and (13), and defining $\xi := \zeta + \sigma \|\hat{x} - \bar{x}\|_\infty$, we obtain

$$\|\tilde{x} - \hat{x}\|_\infty \leq (\eta \|\hat{x}\|_\infty + \xi \|\hat{u}\|_1 \|\bar{x}\|_\infty) \|A - \hat{A}\|_\infty$$

$$+ (\eta + \xi \|\hat{u}\|_1) \|b - \hat{b}\|_\infty + (\eta \|\hat{x}\|_\infty + \xi \|\bar{x}\|_\infty) \|p - \hat{p}\|_1 ,$$

which completes the proof.

The bound given by Theorem 5 involves the dual variables $\hat{u}$. In some cases this may be quite acceptable, if we know some convenient bound on $\|\hat{u}\|_1$. However, if we have no such information then the presence of $\hat{u}$ may be unsatisfactory. In order to eliminate the dependence of the error bound on these variables we must find some way of bounding them in terms of the primal variables and the other parameters of the problem.

One way to do this would be to apply Theorem 1 to bound the individual $\hat{u}$'s in terms of $\mu([\hat{A}\ \hat{b}])$, then use Theorem 3 to bound the latter quantity in terms of $\mu([A\ b])$; this could be done if the augmented matrix $[A\ b]$ satisfied the PLI condition. It is not hard to show that if the system of inequalities $Ax \leq b$ is consistent, then $[A\ b]$ satisfies the PLI condition if and only if $Ax \leq b$ is superconsistent (that is, $A\bar{y} < b$ for some $\bar{y} \in \mathbf{R}^n$). In the latter case, it is simpler to bound the dual variables directly, and that is what we shall do in the next theorem.

THEOREM 6: Let ϵ be a positive real number. If there is a $\bar{y}$ such that $A\bar{y} \leq b - \epsilon e$, then for any $p \in \mathbf{R}^n$ and any primal and dual optimal solutions $(\bar{x}, \bar{u})$ of the program $\min\{p^T x \mid Ax \leq b\}$, we have $\|\bar{u}\|_1 \leq \epsilon^{-1} p^T(\bar{y} - \bar{x})$.

PROOF: Since $\bar{x}$ and $\bar{u}$ are primal and dual optimal for the given program, we have

$$\bar{u}^T A + p^T = 0,$$

$$\bar{u}^T b + p^T \bar{x} = 0,$$

and

$$\bar{u} \geq 0 .$$

Multiplying the first equation by $\bar{y}$ and subtracting it from the second, we find that $\bar{u}^T(b - A\bar{y}) = p^T(\bar{y} - \bar{x})$. Thus $\epsilon \|\bar{u}\|_1 = \bar{u}^T(\epsilon e) \leq \bar{u}^T(b - A\bar{y}) = p^T(\bar{y} - \bar{x})$, or $\|\bar{u}\|_1 \leq \epsilon^{-1} p^T(\bar{y} - \bar{x})$, as was to have been shown.

With this theorem, we can now prove the main result of the paper. In what follows, subsc. pts indicate membership in a sequence (not components).

THEOREM 7: Suppose the program $\min\{p^T x \mid Ax \leq b\}$ is solvable; suppose further that the system $Ax \leq b$ is superconsistent. Let $\{A_n\}$, $\{b_n\}$, and $\{p_n\}$ be sequences with $A_n \to A$, $b_n \to b$, and $p_n \to p$. Suppose a bounded sequence $\{x_n\}$ exists with the property that for each n, x_n solves $\min\{p_n^T x \mid A_n x \leq b_n\}$. Then there is a constant τ such that for every n,

$$d_n \leq \tau(\|A - A_n\|_\infty + \|b - b_n\|_\infty + \|p^T - p_n^T\|_\infty),$$

where d_n is the distance from x_n to the solution set of $\min\{p^T x \mid Ax \leq b\}$.

PROOF: We shall let d_n be the ℓ_∞ distance; the theorem is of course true for distance in any other norm, since all are equivalent to $\| \|_\infty$.

Since $Ax \leq b$ is superconsistent, we can find a real number $\epsilon > 0$ and a vector $\bar{y} \in \mathbb{R}^n$ such that $A\bar{y} - b \leq -\epsilon e$. Also, since $A_n \to A$ and $b_n \to b$, there is some integer N such that for all $n > N$ we have $A_n\bar{y} - b_n \leq -\frac{1}{2}\epsilon e$. Let $\|x_n\|_\infty \leq B$ for all n; then if $\{u_n\}$ is any sequence of dual solutions corresponding to $\{x_n\}$ we must have by Theorem 6, for all $n \geq N$,

$$\|u_n\|_1 \leq 2\epsilon^{-1} p_n^T(\bar{y} - x_n) \leq 2\epsilon^{-1} C(\|\bar{y}\|_\infty + B),$$

where C is a bound on $\|p_n^T\|_\infty$ (the sequence $\{p_n\}$, being convergent, must be bounded). Now let $\bar{x}$ solve $\min\{p^T x \mid Ax \leq b\}$, and apply Theorem 5

with $\hat{x} = x_n$. For $n \geq N$, we can establish uniform bounds for the quantities appearing in that theorem as follows (note that η, ξ, α, β, and γ all depend on n, though for simplicity we do not subscript them):

a. θ, ζ and σ are constants.

b. $\eta := \zeta + \theta \|\bar{x} - x_n\|_\infty \leq \zeta + \theta(\|\bar{x}\|_\infty + B)$, and ξ can be similarly bounded.

c. $\alpha := \eta \|x_n\|_\infty + \xi \|u_n\|_1 \|\bar{x}\|_\infty \leq \eta B + 2\xi\epsilon^{-1}C(\|\bar{y}\|_\infty + B)\|\bar{x}\|_\infty$;

β and γ can be bounded in the same way.

Thus, since N is finite we can find some constant τ such that for every n we have $\max(\alpha, \beta, \gamma) \leq \tau$, and therefore if $\tilde{x}$ is as in Theorem 5,

$$d_n = \|\tilde{x} - x_n\|_\infty \leq \tau(\|A - A_n\|_\infty + \|b - b_n\|_\infty + \|p^T - p_n^T\|_\infty),$$

as was to have been shown.

We shall give two simple examples here to show how Theorem 7 (but not Theorem 5) can fail if either the condition of superconsistency or that of the boundedness of $\{x_n\}$ is dropped. The first example is a sequence of programs over $\mathbf{R}$ with the following parameters: $A_n = \begin{bmatrix} -1 \\ -1/n \end{bmatrix}$, $b_n = \begin{bmatrix} 0 \\ -1/n \end{bmatrix}$, $p_n = [1]$. These programs satisfy the boundedness requirement ($x_n = 1$ for each n) but the limit program is not superconsistent. It is easily verified that $d_n = 1$ for each n, so the conclusion of Theorem 7 does not hold. Theorem 5

is still valid, although in a useless form since the dual variables are given by $u_n = [0 \quad n]$, and are thus unbounded. This example also illustrates the fact that it is not sufficient to assume in place of superconsistency that the feasible region $\{x \mid Ax \leq b\}$ has an interior.

The second example consists of a sequence of programs over R^2, with

$$A_n = [-1/n \quad -1], \quad b_n = [0], \quad p_n = [1/n \quad 1].$$

If we let $x_n = [n \quad -1]^T$, we again have $d_n = 1$ for each n. As before, Theorem 5 holds but provides no useful information since $\{x_n\}$ is unbounded. If, on the other hand, we had taken $x_n = [1 \quad -1/n]^T$, then $\{x_n\}$ would have been bounded and Theorem 7 would have been applicable. It is perhaps worthwhile to point out here that Theorem 7 does not require that all sequences of solutions to the perturbed programs be bounded, but only that at least one bounded sequence exist.

It should be noted that instead of requiring $\{x_n\}$ to be bounded in Theorem 7, we could have allowed $\|x_n\|_\infty$ to appear explicitly in the bound for d_n; then, if one had some growth condition on $\|x_n\|_\infty$ and if A_n, b_n and p_n converged quickly enough, one might still show that the d_n must converge to zero at a certain rate. Also, we could have incorporated a continuous perturbation into A, b, and p, instead of using sequences (although then one has to be careful to maintain superconsistency). However, we have not thought it necessary to carry out these details here.

REFERENCES

1. C. Berge: Topological Spaces. New York: Macmillan, 1963.

2. G. B. Dantzig, J. Folkman and N. Shapiro: On the Continuity of the Minimum Set of a Continuous Function. J. Math. Anal. Appl. 17, 519-548 (1967).

3. J. P. Evans and F. J. Gould: Stability in Nonlinear Programming. Operations Research 18, 107-118 (1970).

4. E. Isaacson and H. B. Keller: Analysis of Numerical Methods. New York: Wiley, 1966.

5. O. L. Mangasarian: Nonlinear Programming. New York: McGraw-Hill, 1969.

FINITE ELEMENT ANALYSIS OF BURIED CYLINDERS

J. L. Kirkland
Nuclear Weapons Effects Division
US Army Engineer Waterways Experiment Station
Vicksburg, Mississippi

FOREWORD. This paper was prepared by Mr. J. L. Kirkland of the Nuclear Weapons Effects Division, US Army Engineer Waterways Experiment Station (WES), for presentation at the Army Numerical Analysis Conference, 21 April 1971. The experimental data used was obtained at WES during the period August 1969 through January 1970, and the analysis is based primarily on work accomplished by Mr. Kirkland during the period September 1969 through January 1971 in connection with his dissertation for Mississippi State University. This research effort was sponsored by the Defense Atomic Support Agency (DASA) under project SC 210 with the Mississippi State University Computing Center providing the computational support. The paper has been approved for publication by the Office, Chief of Engineers, U. S. Army.

Director of WES During the preparation of this paper was COL Ernest D. Peixotto, CE; Technical Director was Mr. F. R. Brown.

Contents

Conversion Factors, British to Metric Units of Measurement

British units of measurement used in this report can be converted to metric units as follows:

Multiply	By	To Obtain
inches	25.4	millimeters
inch-pounds	0.1129848	meter-newtons
pounds	4.448222	newtons
pounds per square inch	6.894757	kilonewtons per square meter

The remainder of this paper has been reproduced photographically from the author's manuscript.

Summary

The objective of this study was to evaluate a finite element method for predicting the response of a cylindrical shell buried in sand and subjected to a static surface pressure. Test results for cylinders of three thicknesses (3/8, 1/4, and 1/8 in.) were compared with the response calculated using the finite element computer code.

Comparison of the computer program results with the experimental results revealed a wide discrepancy between the two. Analysis of the results indicated that it is doubtful that any predicting scheme which uses a homogeneous soil model will be successful since values of moment and thrust which develop in stiff cylinders are very sensitive to the immediate soil environment of the cylinder and it is difficult to obtain a uniform backfill in this region even in the laboratory.

Based on the results reported, it is not possible to make a general assessment of the worth of the finite element approach to the analysis of buried structures. However, the findings are valuable in that they:

a. Point out the need to know the characteristics of the soil backfill in order to use the finite element analysis to predict the response of buried cylinders.

b. Serve as a guide to future extension of the experimental program by pointing out the difficulty of constructing a uniform backfill even in the laboratory.

c. Bring out the increased significance placed on backfill characteristics due to the stiff cylinders used in the experimental program.

d. Show the danger of extrapolating design procedures derived from flexible cylinder data.

FINITE ELEMENT ANALYSIS OF BURIED CYLINDERS

Introduction

Background

1. In the design of buried conduits, ways and methods are constantly being sought to reduce the cost of construction. To accomplish this and still ensure that the conduits function properly requires a means of predicting the load distribution on the buried structures.

Objective

2. The objective of this study was to evaluate a finite element method for predicting the response of a cylindrical shell buried in sand and subjected to a static surface pressure.

Test geometry

3. For the test configuration illustrated in fig. 1, the testing arrangement should provide a plane strain condition for the soil and this is used in the finite element idealization.

4. Fig. 2 shows the thickness of the three cylinders treated in the analysis (3/8, 1/4, and 1/8 in.*). The detailed diagram of fig. 3 indicates the arrangement of these test specimens to produce a plane stress configuration for each cylinder.

Soil model

5. In formulating any equivalent continuum model for a soil medium, one is always faced with the problem of the limitations on the type of tests which can be performed with reasonable accuracy and must be aware of the geometry associated with these tests. The two tests used in this approach are (a) the constrained modulus test shown in fig. 4 and (b) the triaxial compression test illustrated by fig. 5.

6. The information contained in the experimental curve of fig. 4 is input to the finite element computer code in tabular form by partitioning the curve into five straight-line segments.

7. Curves shown in fig. 5 are used to determine the constants in the relation

---

* A table of factors for converting British units of measurement to metric units is presented on page vii.

$$S_2 = -\left(\frac{2\mu}{1 + \frac{2\mu}{\sqrt{\bar{\sigma}}}\sqrt{-I_2'}}\right)^2 I_2' \qquad (1)$$

where μ is the "initial" shear modulus, $\bar{\sigma}$ is associated with a "yield" surface, and S_2 and I_2' are the second invariants of the deviators stress and strain, respectively. Using equation 1 to define the shear modulus

$$G = \frac{2\mu}{1 + \frac{2\mu}{\sqrt{\bar{\sigma}}}\sqrt{-I_2'}} \qquad (2)$$

With the aid of equation 2 and the information of fig. 4, an incremental constitutive relation of the form

$$\begin{Bmatrix} \sigma_x \\ \sigma_y \\ \sigma_{xy} \end{Bmatrix} = \begin{bmatrix} K + \frac{4}{3}G & K - \frac{2}{3}G & 0 \\ K - \frac{2}{3}G & K + \frac{4}{3}G & 0 \\ 0 & 0 & 2G \end{bmatrix} \begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_{xy} \end{Bmatrix}$$

is used for each soil element in the finite element model. In this relation, G is defined by equation 2 and K is the bulk modulus.

Analysis and Comparison of Results

8. The approach used in this study to include the physical nonlinearities of the soil medium is that of adding a succession of linear incremental problems. For the incremental problems, each element is assigned linearly elastic constants based on the current states of stress and strain that exist in that element. As one should expect, this type of detailed analysis is costly computerwise. For instance, in the following results, 50 incremental problems were used for each cylinder configuration, requiring approximately 21 minutes of Univac 1106 computer time.

9. The boundary value problem described in fig. 1b is partitioned

into 314 elements using 347 node points as shown in fig. 6. The cylinders are represented in the finite element idealization using two elements to span their wall thicknesses.

10. Fig. 7 shows comparisons of the calculated values of the change in vertical diameter with the experimental values for the 3/8- and 1/8-in. cylinders. The solid line is the experimental measured change in vertical diameter with respect to static pressure on the soil surface. As can be seen, the calculated values fall consistently below those measured in the experiment.

11. Fig. 8 shows measured and calculated moments for the 3/8-in. cylinder. The pattern of variation for the other two cylinder sizes was similar. The discrepancies in the graphs are consistent with the deviations between calculated and measured deflection displayed in fig. 7. Measured and calculated thrusts for all three cylinder sizes are shown in figs. 9-11. The experimental thrust calculation involves taking the difference of two recorded strain gage measurements, and this accounts for the erratic variations of the thrust curves. Even though the oscillation in the experimental thrust curves indicates scatter in the data, it is felt that the curves reflect global behavior of the thrust, and that the large deviation of the calculational results is meaningful.

12. The deformed shapes of the cylinders were calculated by subtracting out the rigid body motion of the cylinders (fig. 12). Only the vertical diameter change was measured in the experiment, and this data point is indicated in the figure. The deformation patterns are substantially as one would expect under the loading conditions.

13. Figs. 13-15 are comparisons of the calculated vertical stresses at the crown with the calculated horizontal stresses at the spring line in the soil elements adjacent to the cylinders. These graphs indicate the relative stiffness of the cylinder to that of the soil: the 3/8-in. cylinder giving σ_{yy} at the crown greater than σ_{xx} at the spring line; the 1/4-in. cylinder showing the two stress components to be relatively close together; the 1/8-in. cylinder having the σ_{yy} at the crown less than σ_{xx} at the spring line. In addition, these curves are indicative of the lateral load developed on the cylinder in the finite element analysis of the

problem. The previous discussion concerning the experimental thrust curves shows that this lateral load did not develop in the experiments, especially on the stiffer cylinders.

14. In fig. 16, the solid lines give the stress components measured in the soil in a typical test as a function of the surface pressure. The points on these graphs represent the calculated stress components at the same location. The agreement between the calculated and measured values is an indication that the equivalent continuum model is representative of the soil insofar as the free-field stress components are concerned.

15. The calculated deflections of the soil surface for the three cylinder configurations are shown in fig. 17. These curves are also indicative of the relative stiffness of the cylinders as compared to the soil model.

16. Figs. 18 and 19 display the stress components for the entire soil field for the 3/8-in. cylinder with an applied surface pressure of 500 psi. The measured values are shown in parentheses at the location of the measuring gage. These are the peak values taken from the curves such as that shown in fig. 16.

17. The discrepancies between calculated values and experimental curves indicates a pattern that suggests that the poor agreement between experimental and calculated values results from the use of a single soil model for the sand medium rather than from the type of soil model used in the calculations. The placement of the cylinders and the subsequent burial procedures apparently produce a loose zone of sand just below the spring line and adjacent to the cylinders. This explanation of the experimental results is discussed in more detail in the next section.

Conclusions and Recommendations

18. Although the geometry involved in this study is not complex, the nature of the problem was such that it presented a crucial test for a soil-structure interaction code. The stiffness of the cylinders involved varied sufficiently to cover the cases of active arching (the structure carries less load proportionally than the surrounding soil) to passive arching (the structure carries more load proportionally than the surrounding soil).

Thus, the loading conditions of the soil in the vicinity of the cylinders differed significantly from the 3/8-in. cylinder to the 1/8-in. cylinder.

19. Comparison of the computer program results with the experimental results revealed a wide discrepancy between the two. There are several indicators that the environment of the cylinders was different in the model from that of the tests. For instance, the recorded free-field components of stress are in reasonable agreement with the calculated values, but the measured values of thrust at the crown of the 3/8-in. cylinder indicate that it received essentially no lateral support from the soil at these pressure levels. On the other hand, the measured values of thrust at the crown of the 1/8-in. cylinder indicate that it received lateral support from the soil. Fig. 12 indicates the different magnitudes of displacement experienced by the soil in the region of the spring line of the cylinders This suggests that a reduced density of the sand adjacent to the spring line due to placement techniques would cause the cylinders to support a surface load in quite a different manner. The stiff cylinder would deflect very little at the spring line, and thus would derive very little support from the surrounding soil due to the reduced density zone. However, the relatively flexible cylinder would experience a deflection at the spring line that would mask the reduced density zone. Of course, if the compression mode of the cylinder is not mobilized, the load must be carried as moment. Thus, the moment would be greater than expected in such cases. This is apparently the situation that developed in the experiments considered in this analysis.

20. It is doubtful if any predicting scheme which uses a homogeneous soil model will be successful. The values of the moment and thrust which develop in stiff cylinders are very sensitive to the immediate soil environment of the cylinder. This is due primarily to small deformation of the cylinder-soil interface. Thus, irregularities which are masked by the larger deflections associated with the flexible cylinders become extremely important for stiff cylinders. Therefore, the analysis of stiff cylinders is compounded by the fact that a detailed knowledge of the interface zone is needed, and, yet, this is the region about which the least is known. Meaningful measurements in this region are difficult to obtain.

21. One method of attack on such problems would be to consider a region below the spring line and adjacent to the cylinder (the region where the sand placement is most difficult) as a material with properties different from the other sand. By making computer calculations using such a heterogeneous soil model, the influence of a reduced density zone could be varied in subsequent computer calculations in an attempt to obtain the experimental values for moment and thrust.

22. If it is possible to determine a heterogeneous soil model which predicts the high moment and low thrust response as measured in these experiments, the same scheme can be examined for buried structures of other shapes. By extending the investigation to cylinders of larger diameter, perhaps the normal component of stress can be measured around the cylinder, giving an experimental indication of the value of the lateral load on the buried cylinders. Therefore, it is recommended that future effort be directed along this approach.

23. Even though the calculational results differed significantly from the experimental curves, these findings are useful in that they:

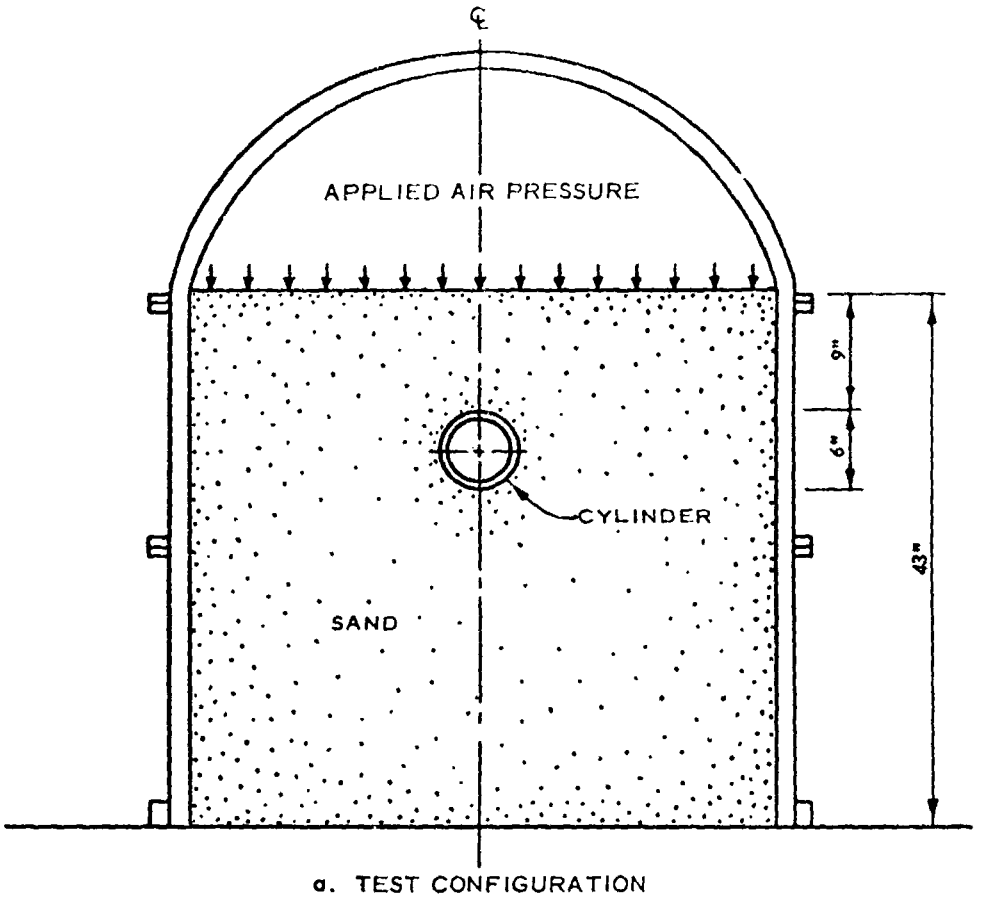
a. Point out the need to know the characteristics of the soil backfill in order to use the finite element analysis to predict the response of buried cylinders.

b. Serve as a guide to future extension of the experimental program by pointing out the difficulty of constructing a uniform backfill even in the laboratory.

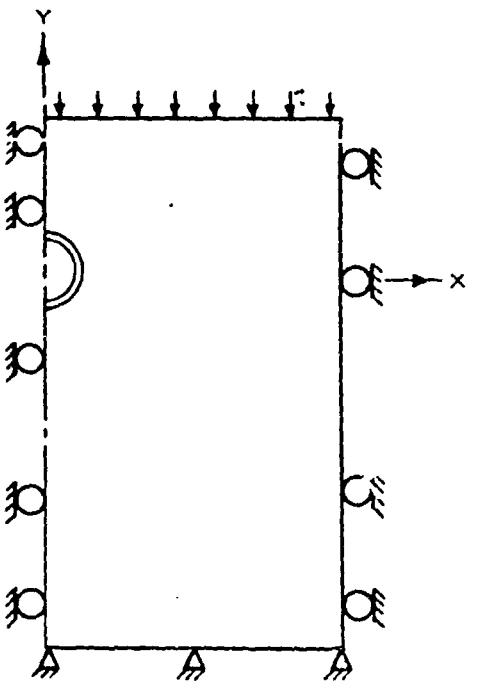
c. Bring out the increased significance placed on backfill characteristics due to the stiff cylinders used in the experimental program.

d. Show the danger of extrapolating design procedures derived from flexible cylinder data.

24. However, it is not possible to make a general assessment concerning the worth of this finite element approach to analyze buried structures from this effort due to the unknown density distribution around the cylinders. Certainly the calculational results cannot be expected to reflect characteristics which are not built into the soil model. In this case, with the stiff cylinders and relatively low surface pressure, it is apparently the conditions at the soil-cylinder interface that dictated the mode of response of the cylinder.



a. TEST CONFIGURATION



b. BOUNDARY CONDITIONS

Fig. 1. Test configuration and corresponding plane boundary value problem

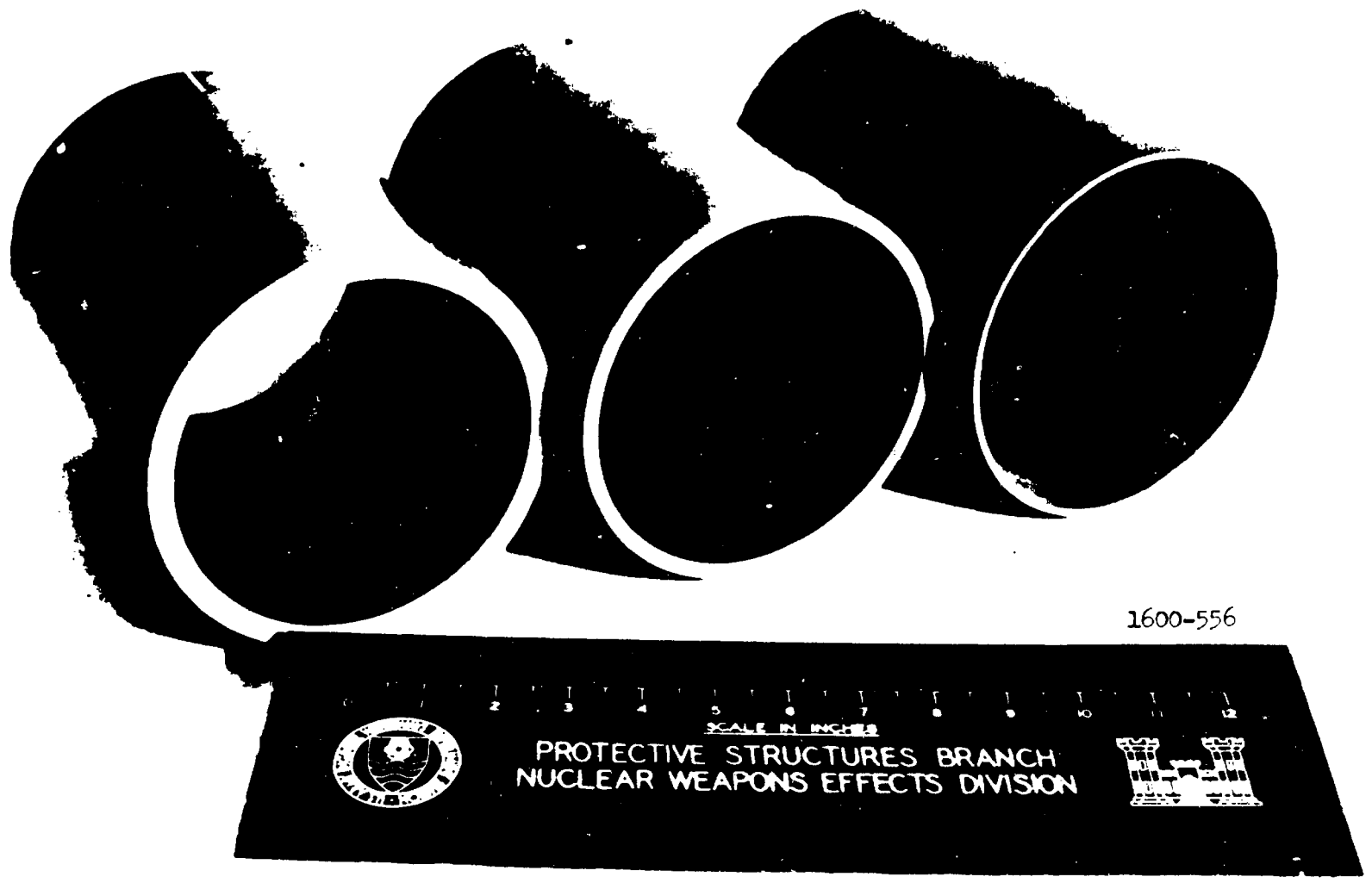
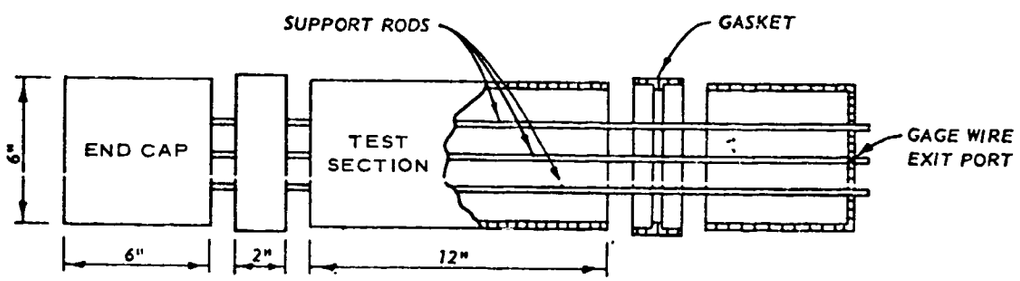


Fig. 2. Comparison of cylinder thicknesses

a. EXPLODED VIEW



b. GEOMETRY

Fig. 3. Exploded view and geometry of test structure

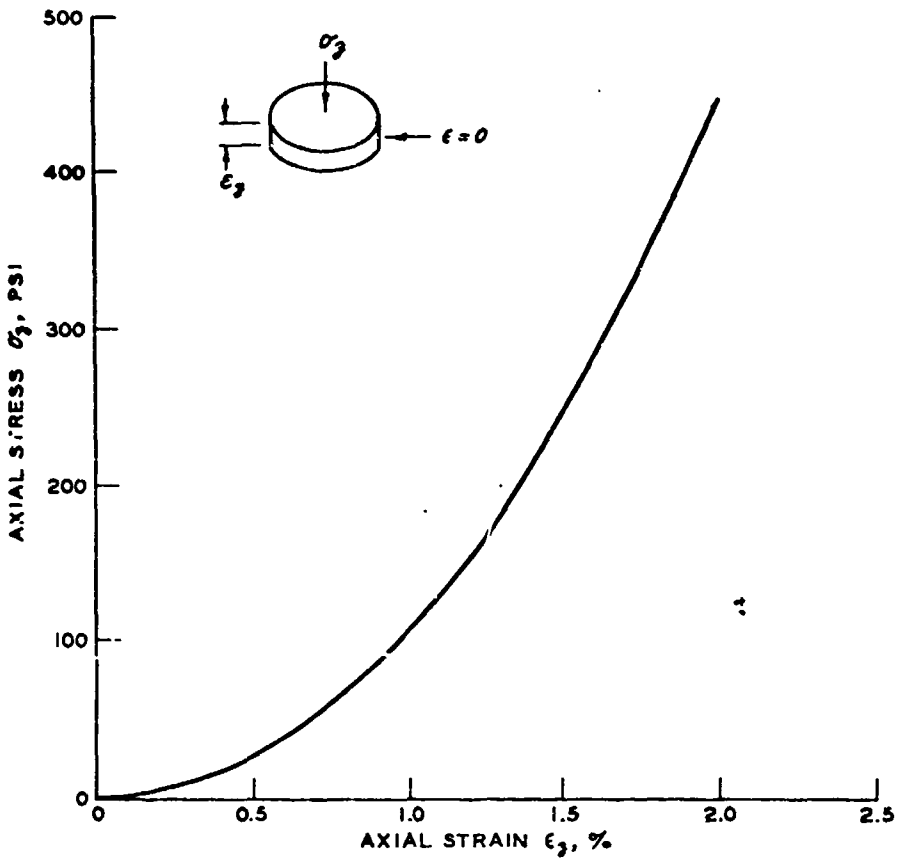


Fig. 4. Constrained modulus test

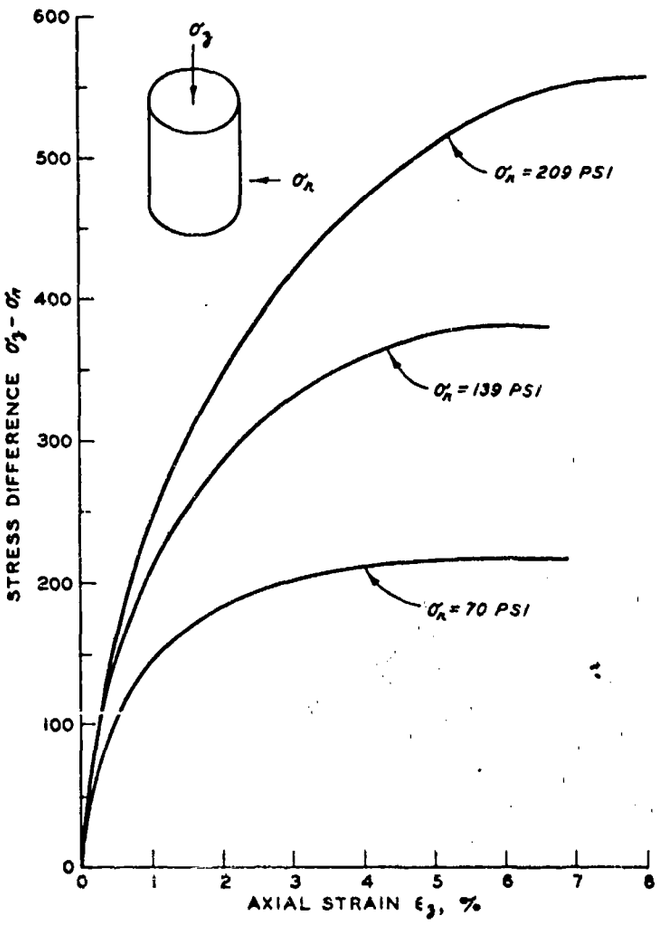


Fig. 5. Triaxial compression test

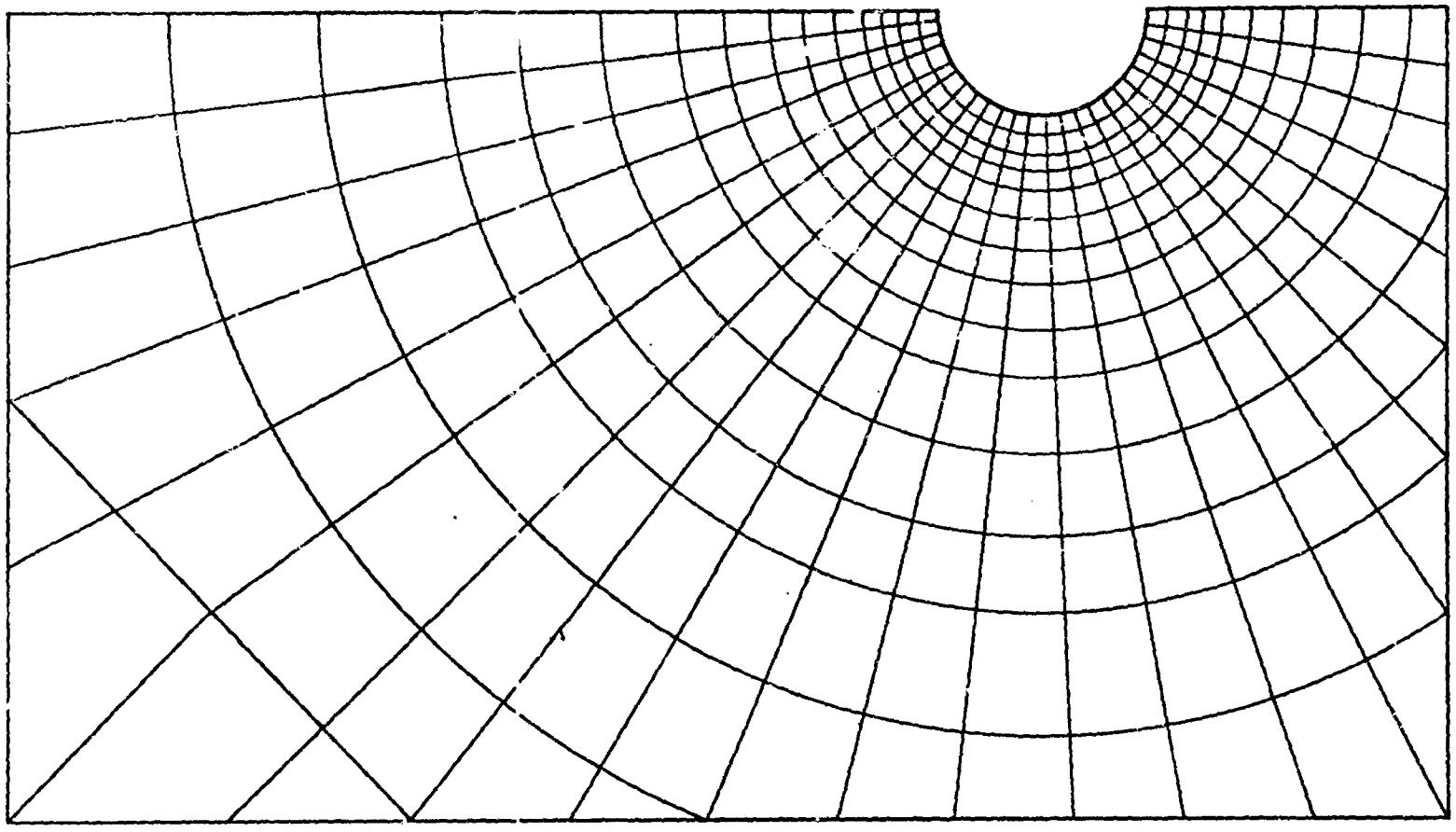
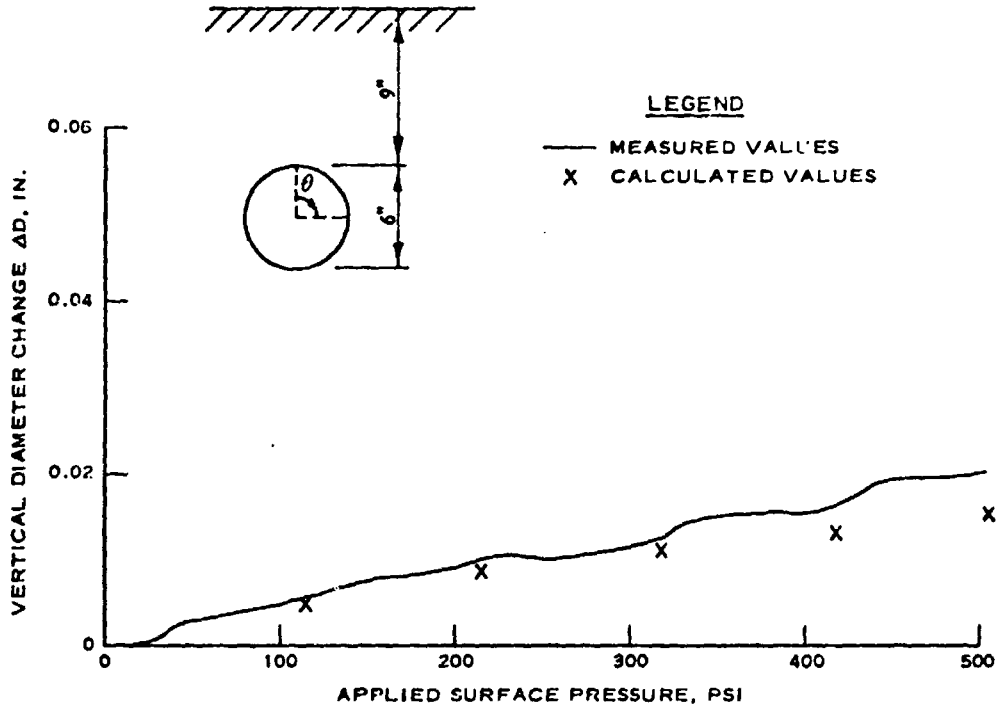
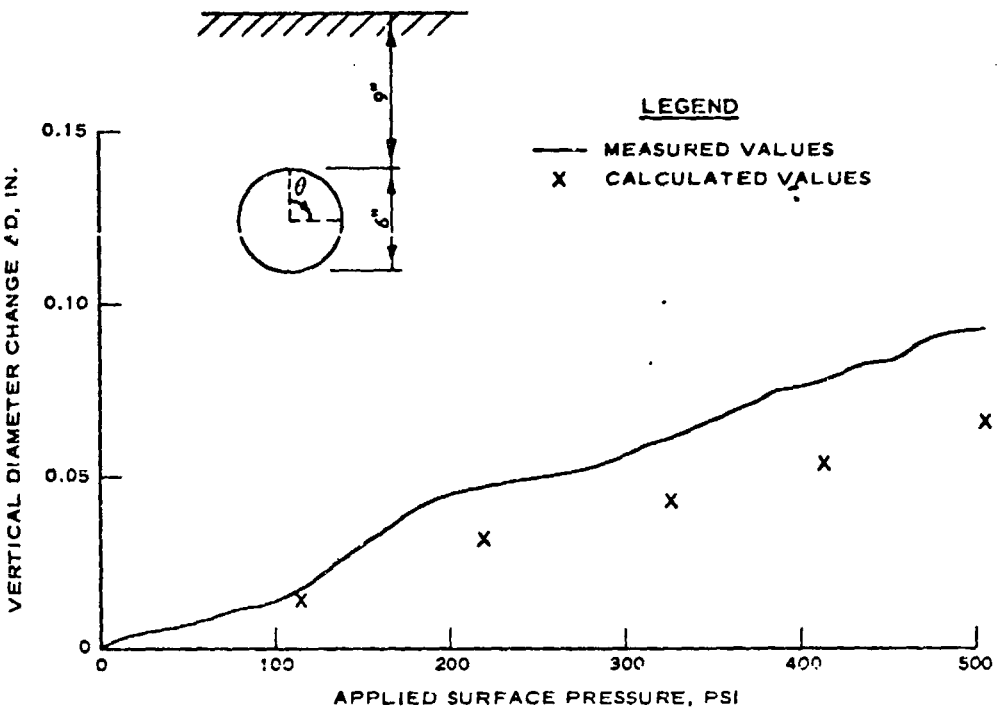


Fig. 6. Finite element idealization of the buried cylinder configuration

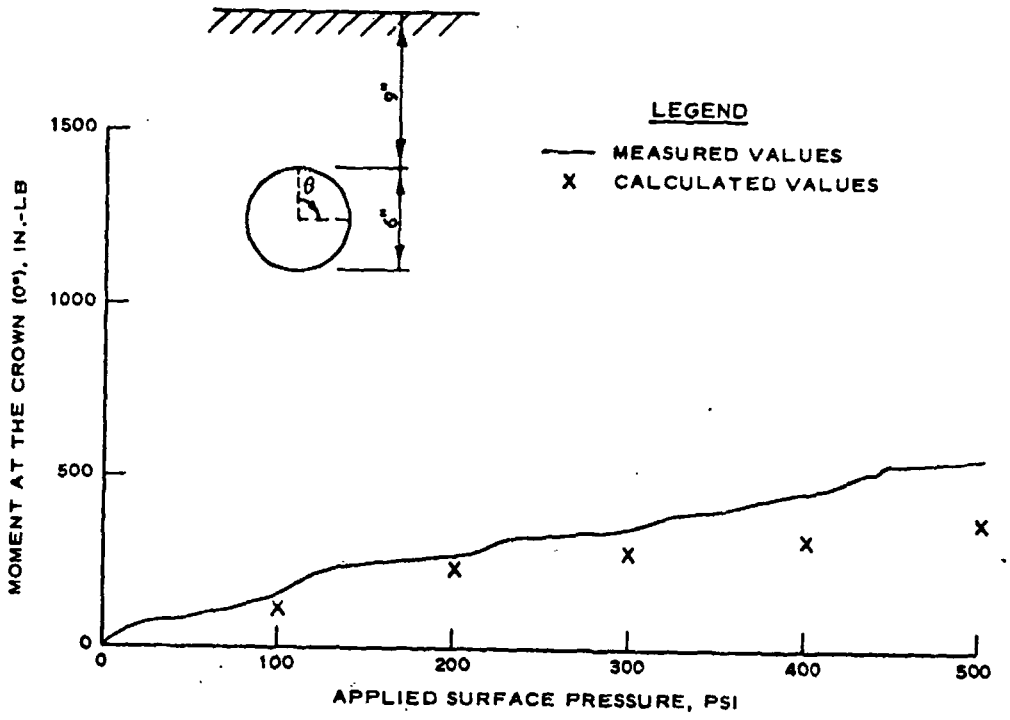


a. 3/8-in. cylinder

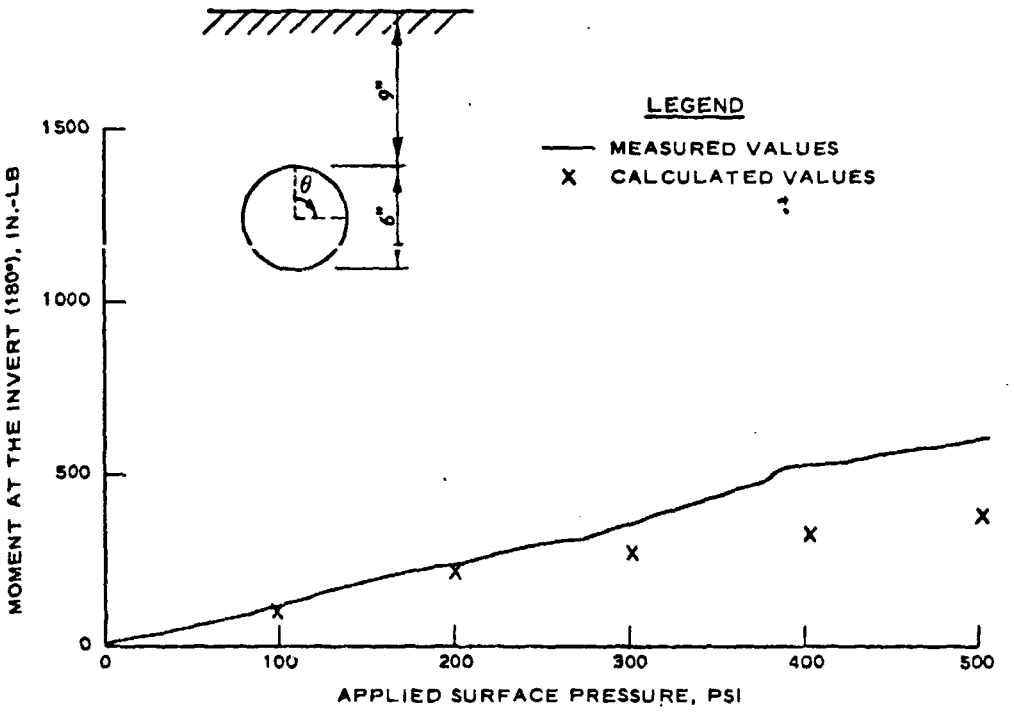


b. 1/8-in. cylinder

Fig. 7. Comparisons of predicted and recorded diameter change

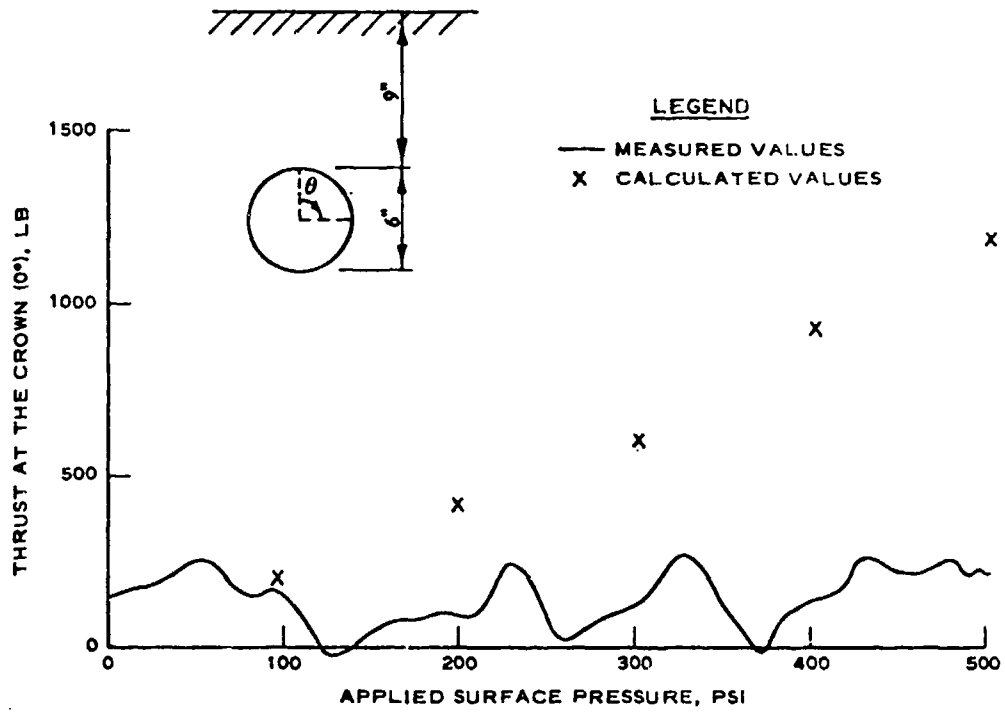


a. Crown

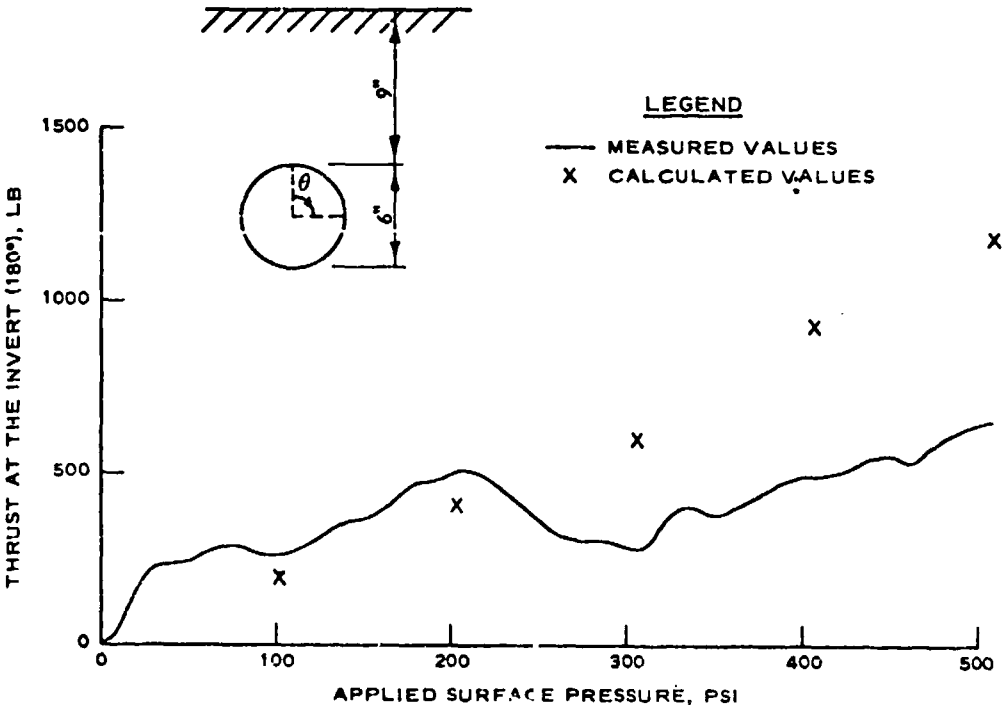


b. Invert

Fig. 8. Variation of moment with applied surface pressure for 3/8-in. cylinder

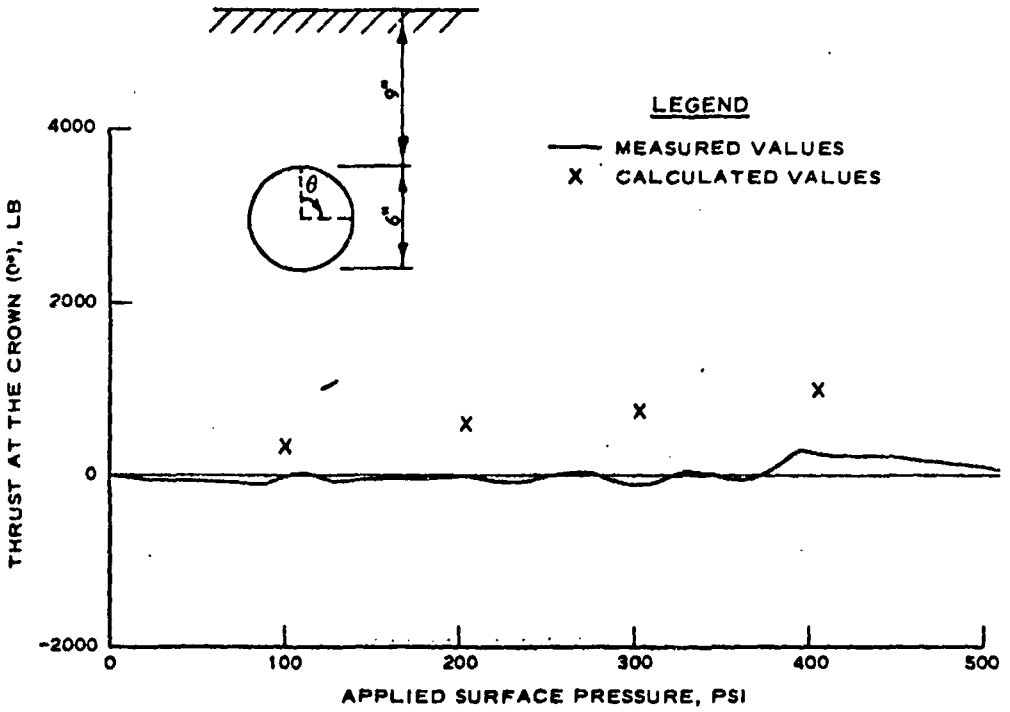


a. Crown

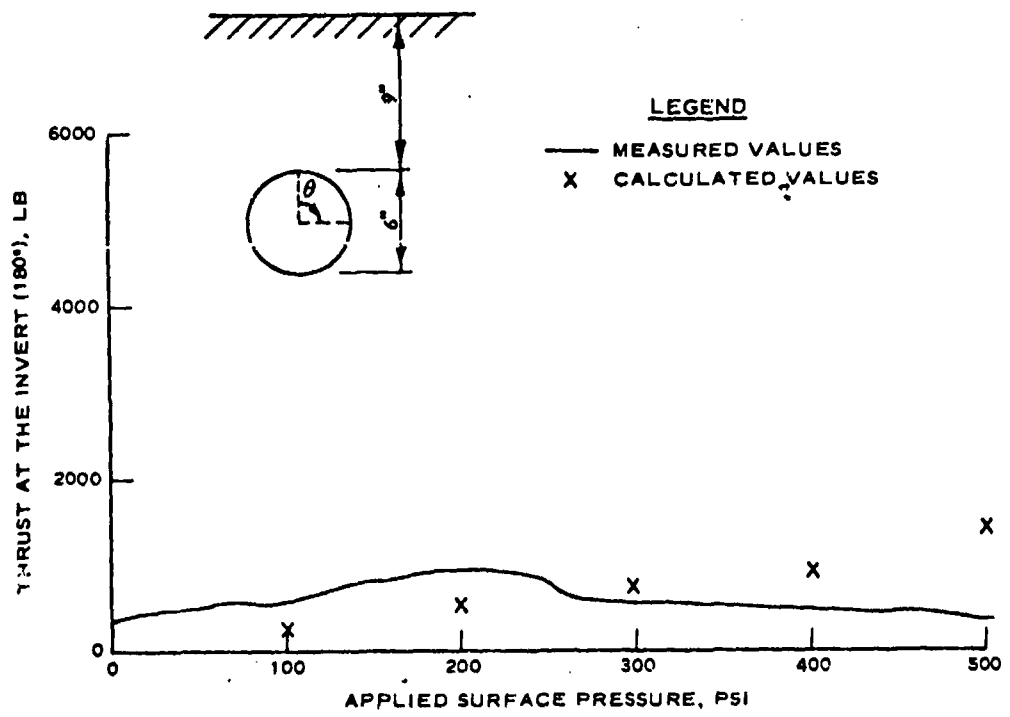


b. Invert

Fig. 9. Variation of thrust with applied surface pressure for 3/8-in. cylinder

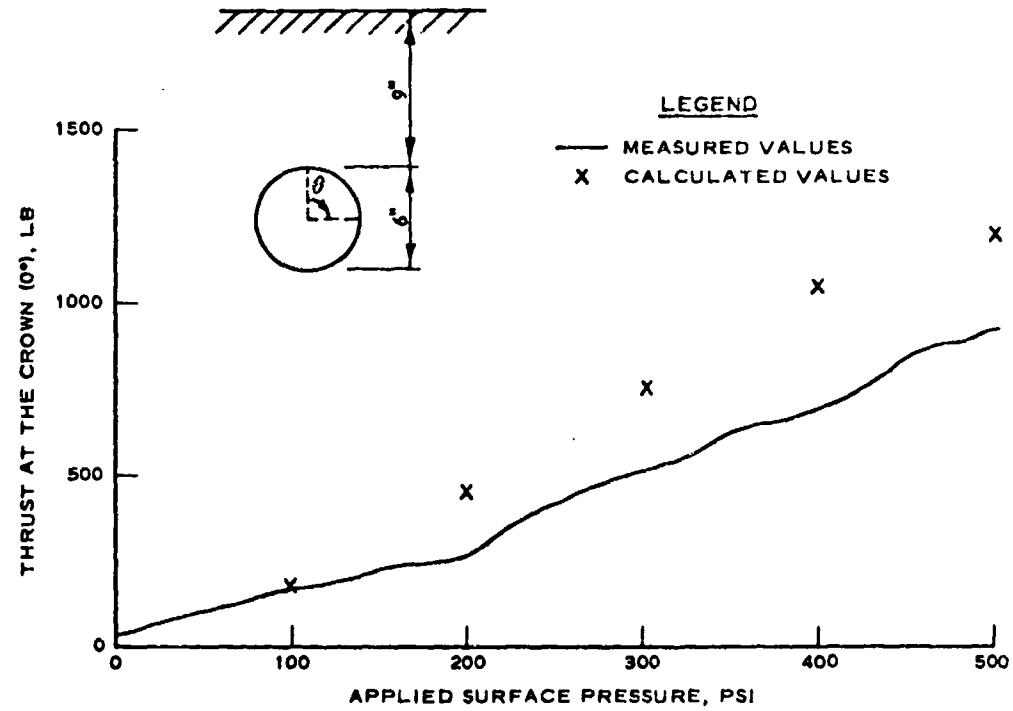


a. Crown

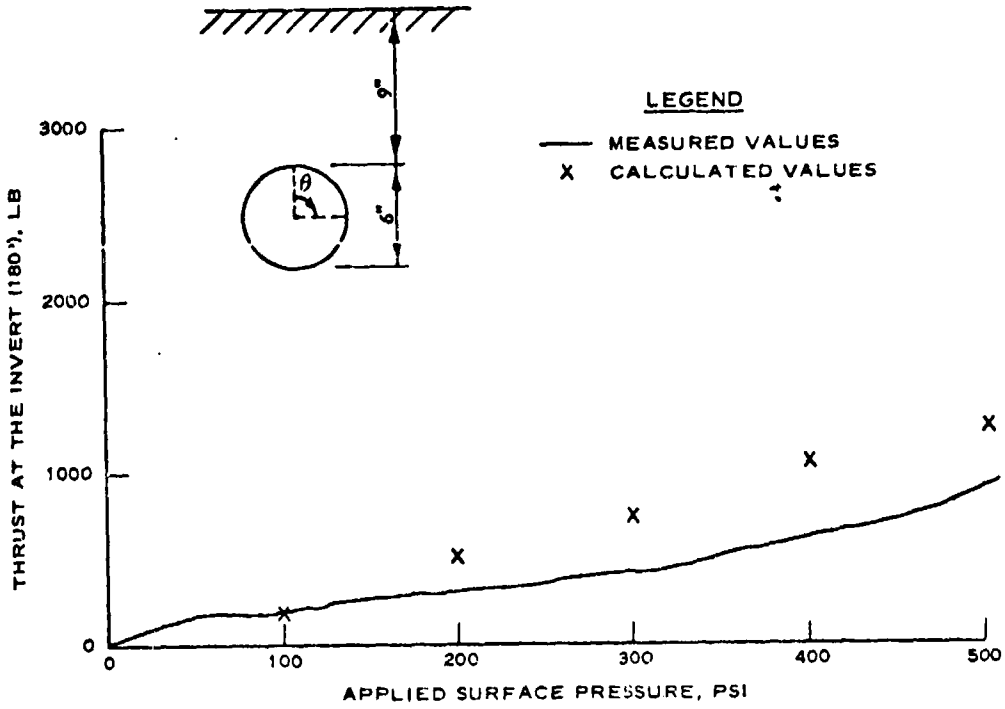


b. Invert

Fig. 10. Variation of thrust with applied surface pressure for 1/4-in. cylinder

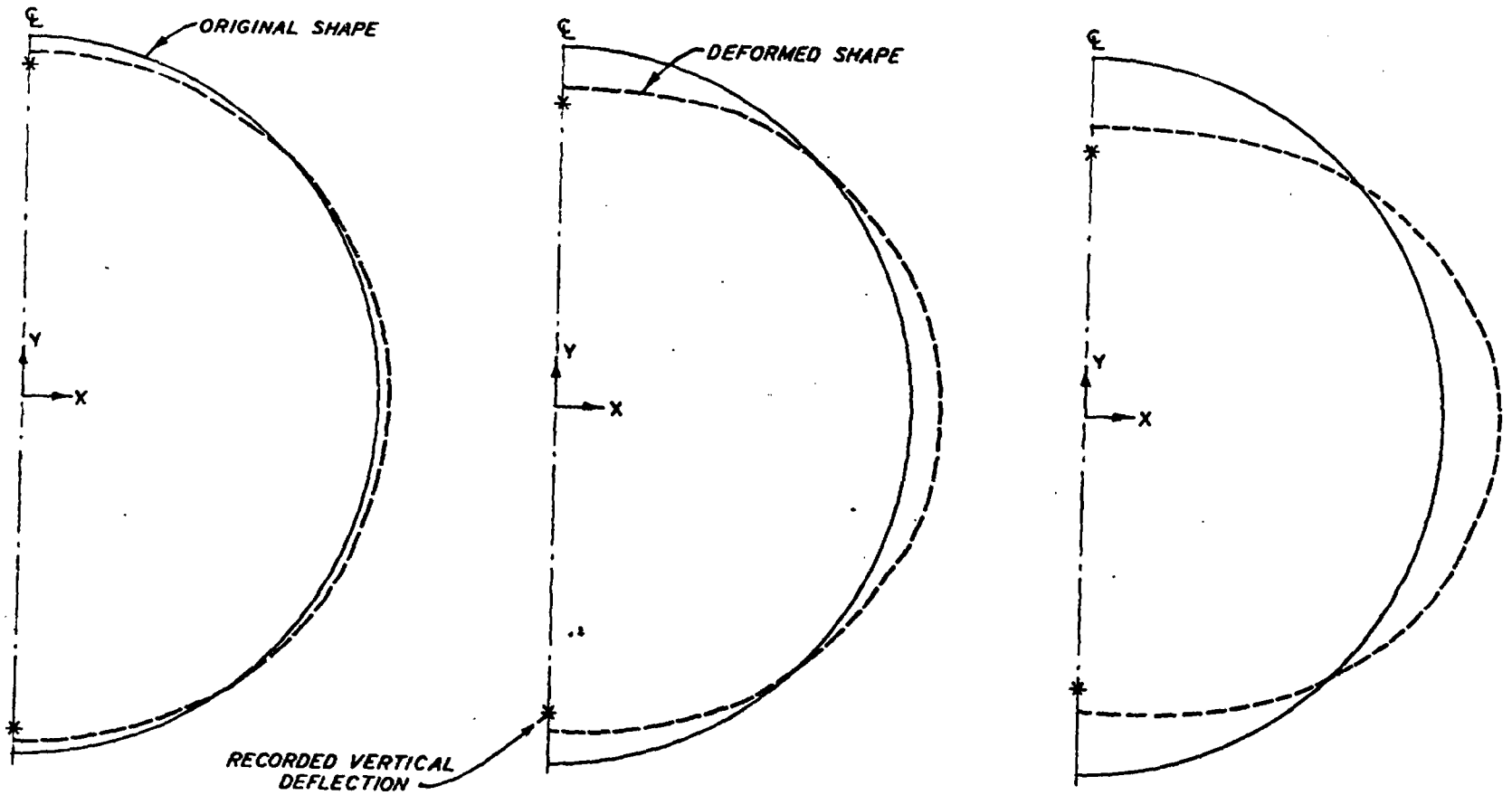


a. Crown



b. Invert

Fig. 11. Variation of thrust with applied surface pressure for 1/8-in. cylinder



a. 3/8-in. cylinder b. 1/4-in. cylinder c. 1/8-in. cylinder

Fig. 12. Calculated deformation patterns

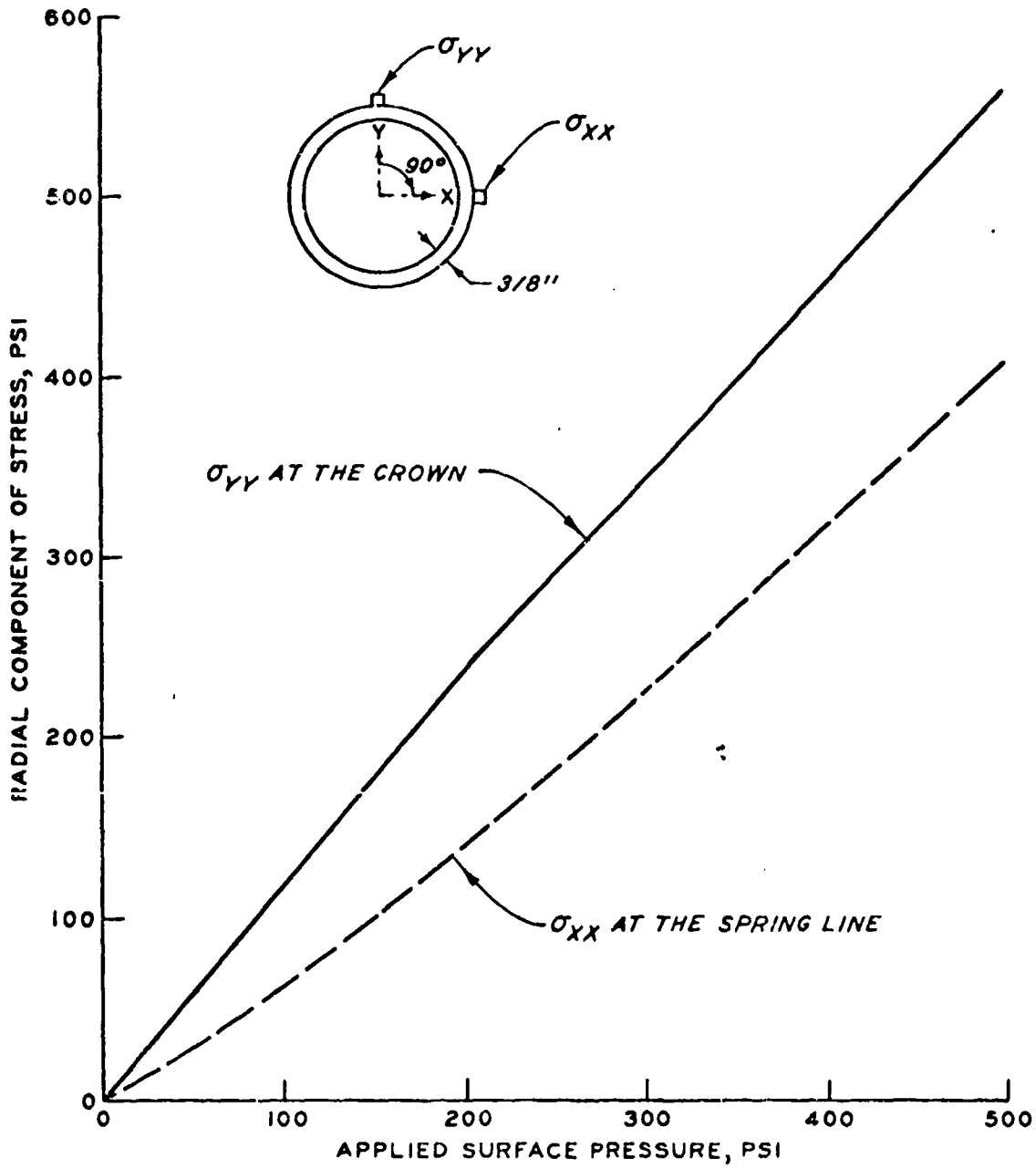


Fig. 13. Comparison of the calculated radial stresses at 0 and 90 deg for the 3/8-in. cylinder

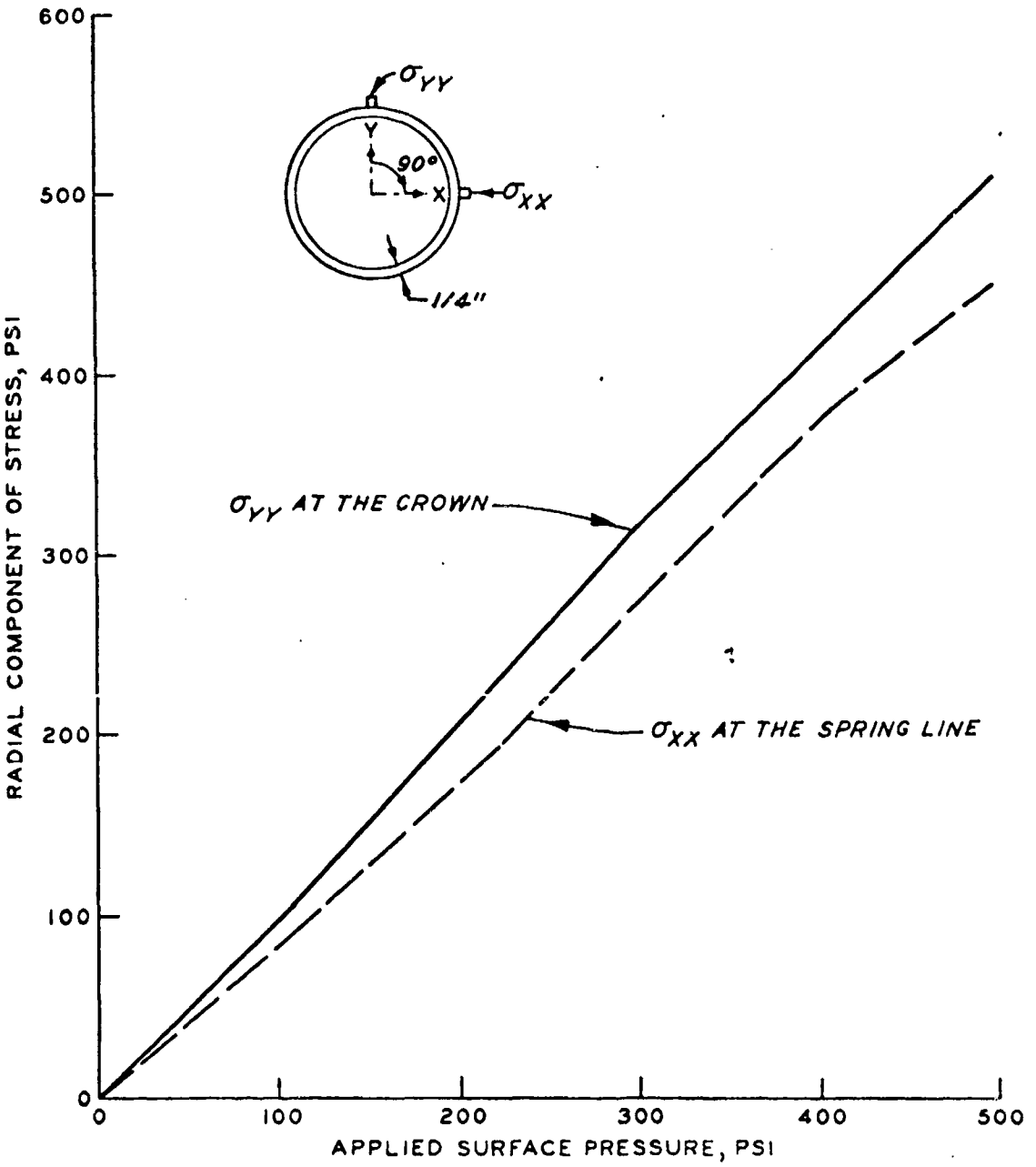


Fig. 14. Comparison of the calculated radial stresses at 0 and 90 deg for the 1/4-in. cylinder

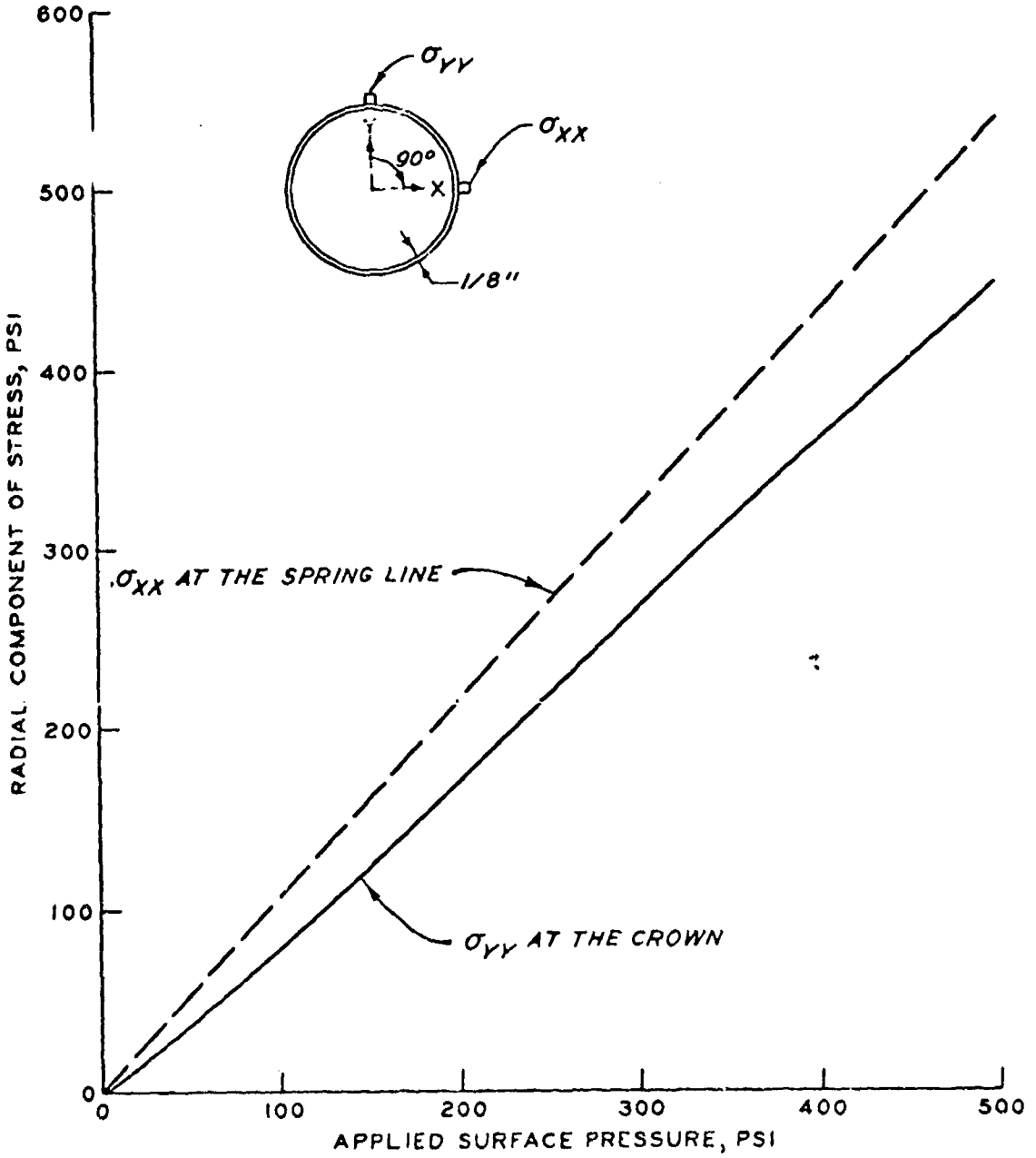


Fig. 15. Comparison of the calculated radial stresses at 0 and 90 deg for the 1/8-in. cylinder

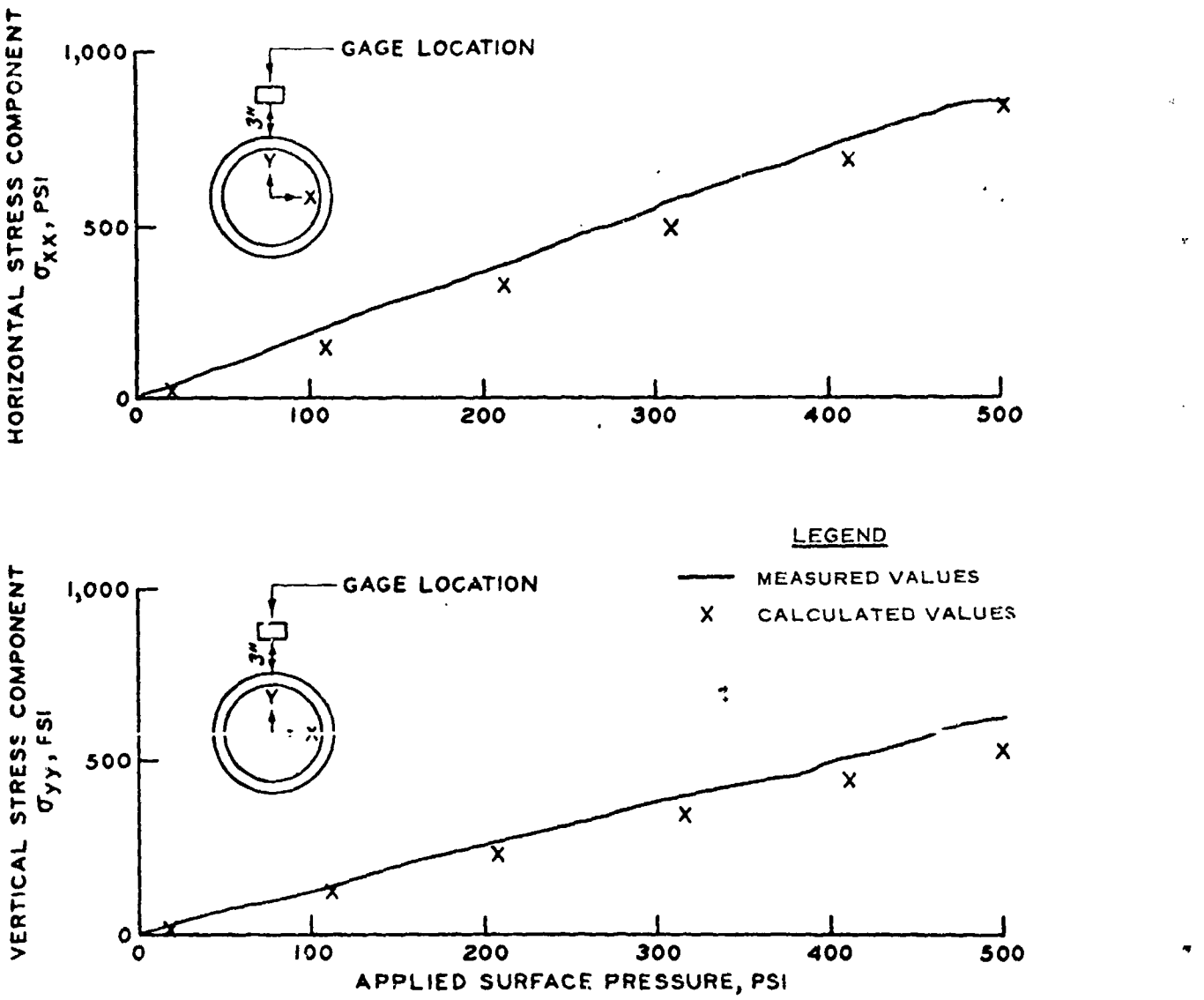


Fig. 16. Comparison of predicted and measured stress components for 3/8-in. cylinder at 3 in. above the crown

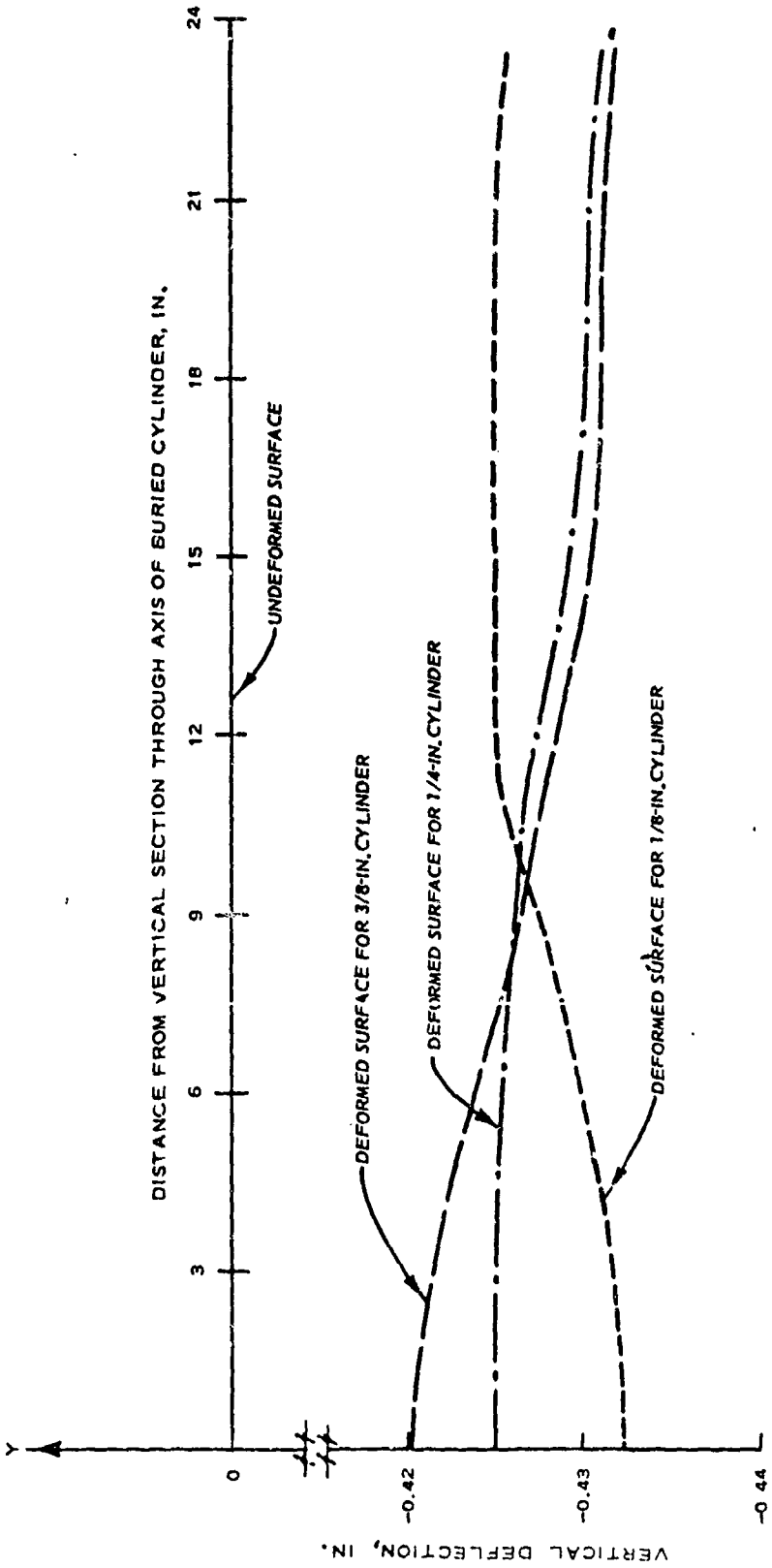


Fig. 17. Calculated soil surface deflections

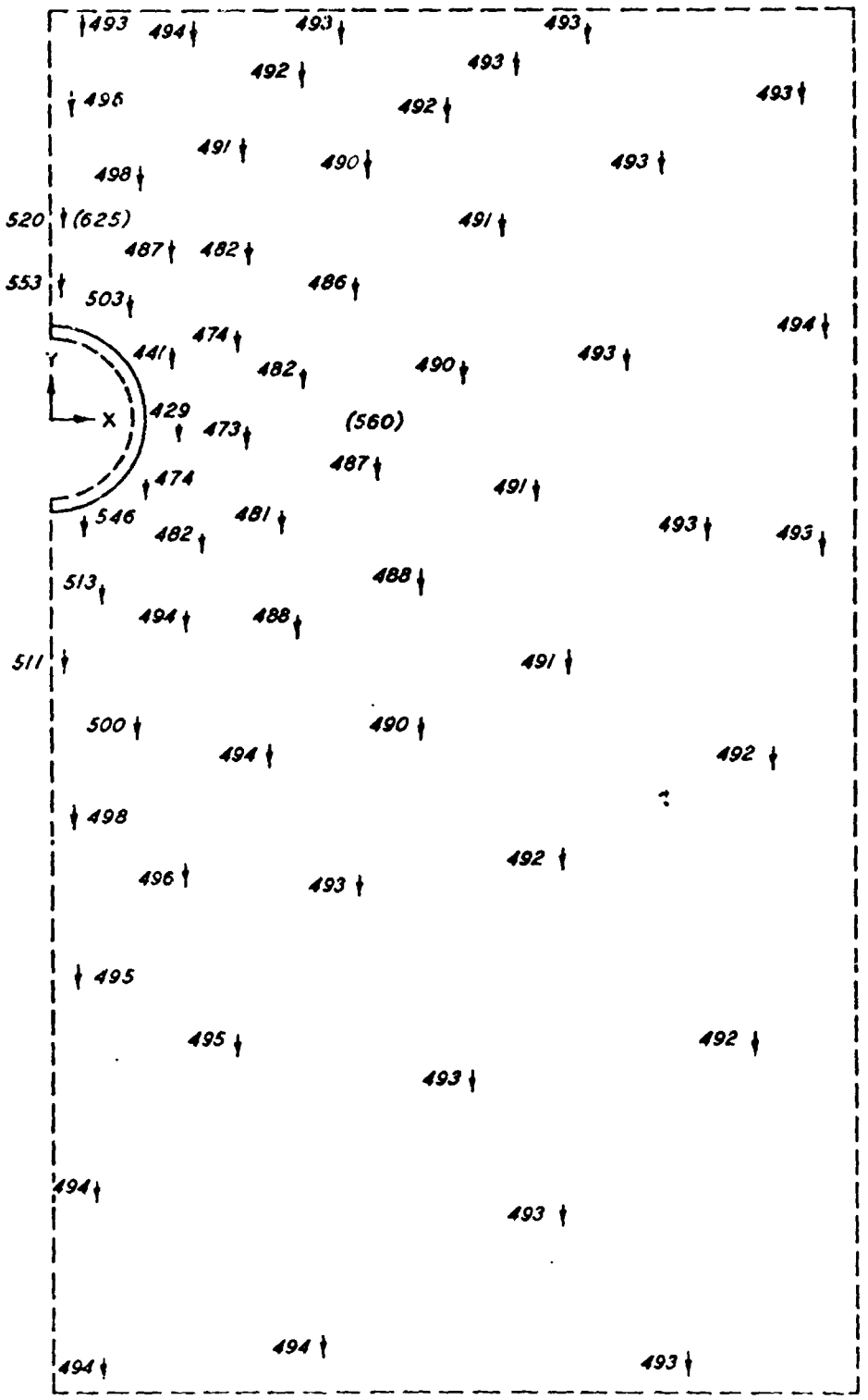


Fig. 18. Calculated distribution of σ_{yy} (psi) in the soil for the 3/8-in. cylinder. (Recorded values are shown in parentheses at gage location)

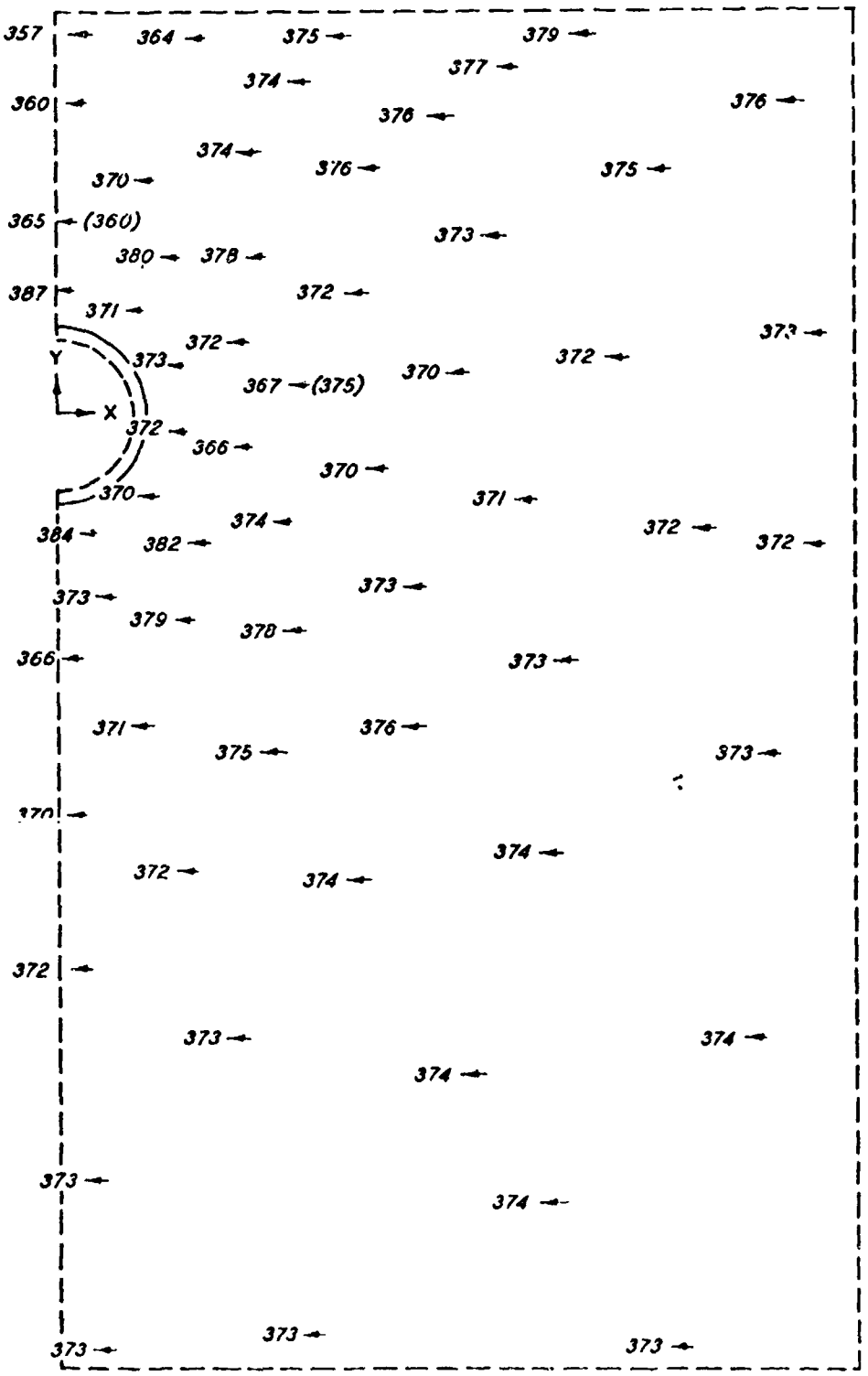


Fig. 19. Calculated distribution of σ_{xx} (psi) in the soil for the 3/8-in. cylinder. (Recorded values are shown in parentheses at gage location)

APPLICATION OF SPLINE INTERPOLATION METHODS TO ENGINEERING PROBLEMS

J. B. Cheek, Jr., N. Radhakrishnan, F. T. Tracy
Computer Analysis Branch
Automatic Data Processing Division
US Army Engineer Waterways Experiment Station
Vicksburg, Mississippi

FOREWORD. This paper was prepared by Mr. J. B. Cheek, Jr., Dr. N. Radhakrishnan, and Mr. F. T. Tracy of the Computer Analysis Branch, Automatic Data Processing Division, U. S. Army Engineer Waterways Experiment Station (WES), for presentation to the 1971 Army Numerical Analysis Conference sponsored by the Army Mathematics Steering Committee and hosted by the Department of Defense Computer Institute. The paper was reviewed and approved for presentation and publication by the Office, Chief of Engineers, U. S. Army.

The work was conducted during the period Dec 1967 to Apr 1971 under the general supervision of Mr. D. L. Neumann, Division Chief. It is based on work for several of the WES technical divisions in connection with a number engineering projects.

During the period in which this paper was prepared, COL Ernest D. Peixotto, CE, was Director of WES; Mr. F. R. Brown was Technical Director.

CONTENTS

The remainder of this paper has been reproduced photographically from the author's manuscript.

Preceding page blank

SUMMARY

This paper was prepared to familiarize practicing scientists and engineers with the cubic spline interpolation technique as a possible tool in curve fitting for computer programs for which more commonly used techniques may be unsuitable or of limited value. The spline technique is compared with more common methods, specifically piece-wise linear and polynomial, and examples of applications of the technique to engineering problems are presented. Appendix A contains the mathematical derivation of the equations defining the spline function, and Appendix B contains a compact FORTRAN fitting and interpolating program.

The interpolating spline curve-fitting technique has three primary advantages:

a. The spline passes through all data points.

b. The first and second derivatives of the spline are continuous at all points.

c. The spline can be easily modified to satisfy new or additional data.

The experience of the Waterways Experiment Station (WES) in applying spline techniques to engineering problems has indicated that these advantages outweigh the additional storage and/or computation time requirements of the technique in many applications.

Since the spline function is required to pass through all data points, erratic derivative behavior may result from experimental error when the data points are numerous and closely spaced. Trial and error methods for smoothing such functions exist, but they are time consuming. WES experience has indicated that acceptable results can generally be obtained by simply selecting a more limited, more widely spaced set of the data points to which to fit the curve.

APPLICATION OF SPLINE INTERPOLATION METHODS TO ENGINEERING PROBLEMS

PART I: INTRODUCTION

Purpose and Approach

1. Many computer programs applied to engineering research and design problems must model the characteristics of nonlinear physical systems. The cubic spline offers outstanding advantages over the interpolating methods commonly used in this class of problems. It is, therefore, the purpose of this paper to point out the shortcomings of several methods and show how spline techniques may be used to advantage. This purpose is approached through discussion and examples in language and subject that are meaningful to the research and design engineer in order to bring to the practicing scientist and engineer an assurance that the cubic spline formulation offers a powerful, practical modeling and interpolating method for use in his computer codes.

Commonly Used Curve-Fitting Techniques

2. Numerical techniques and digital computers are being applied to an increasing number of civil engineering problems. This is principally due to the flexibility afforded by the numerical procedures, in that the design and research engineer can easily specify complicated boundary conditions and use nonlinear material properties. The finite difference and finite element methods are excellent examples of this growing application area. The valid description (modeling) of the nonlinear properties to the computer program is a primary consideration and is the major concern of this paper.

3. Problems of this type require that nonlinear relationships between physical quantities be available to the solution process in either functional or analytical form. This is necessary to answer questions like:

given a strain, what is the stress; or given a water elevation (stage), what is the flow (discharge). Such relationships are normally available only as data points. It is therefore necessary to use some kind of curve-fitting technique in the computer program to represent the physical system, not only at the data points, but in the intervals between data points.

4. There are many curve-fitting methods, but there is no ideal method. Consequently, a major difficulty in developing the solution process is in selecting one curve-fitting technique that is best suited to the problem at hand.

5. Two of the most commonly used curve-fitting techniques are piece-wise linear and polynomial methods, while hyperbolic function and other special purpose representations are occasionally used. There are, however, serious disadvantages in using the linear and polynomial methods. These limitations are presented in the following paragraphs to help the reader appreciate the advantages of the spline method.

Piece-wise linear interpolation

6. Disadvantages. Given a series of data points that, for engineering purposes, "exactly" represent a physical situation, there is strong motivation to use piece-wise linear interpolation between point pairs. This is especially tempting when additional data points are easy to acquire, because the nonlinearity can be modeled (to the extent required) simply by specifying additional points for the nonlinear ("curvy") regions. It appears that the only disadvantage is the additional computer memory required for the points.

7. The serious objection of discontinuous derivatives develops when modeling observed physical behavior and calculating rates of change (derivatives) from the piece-wise linear model. This difficulty caused by discontinuous derivatives is illustrated in the following example.

8. The nonlinear properties of soil in a finite element method (FEM)[1] solution may be represented with a set of stress-strain points, as shown in fig. 1. The solution process requires interpolation of a stress value for any strain value. Also required is the modulus of elasticity for those same strain values (the modulus being dY/dX, the slope of the stress-strain curve). Fig. 1 also includes the plot of modulus versus strain.

Note the abrupt change in modulus as strain progresses from region A to region B. How does this affect the FEM solution process?

9. To answer that question one must first state three characteristics of a typical nonlinear FEM process: (a) the solution procedure involves solving a series of incremental loading problems, (b) during the load cycle, the modulus of each soil element is dependent on the strain at that element, and (c) the strains increase incrementally from an initial value as additional load cycles are taken.

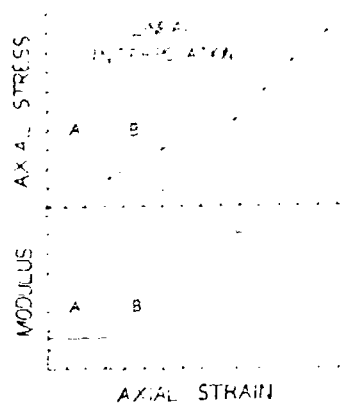


Fig. 1. Piece-wise linear stress-strain approximation

10. Thus, those elements having any value of strain in region A will have one modulus value, while those having any strain value in region B exhibit a different modulus. Consequently, as the element strain progresses from region A to B, the solution process sees an abrupt change in that element's modulus. Such behavior is not characteristic of soil modulus versus strain; i.e., the model for modulus versus strain is obviously invalid. The effect of this abrupt change is instability in the numerical process and questionable results.

11. Summary. Linear interpolation is an easy method to implement in the program. It can accurately model the observed behavior, provided a sufficient number of data points are available; but it fails completely in modeling derivatives and is wasteful of computer memory when stringent limits are placed on allowable errors.

Polynomial interpolation

12. Disadvantages. Because linear interpolation has discontinuous first derivatives, researchers have turned to higher (N^{th}) degree polynomials which have the much desired continuity of their first through $(N - 1)^{th}$ derivatives. In so doing they discovered difficulties and deficiencies associated with polynomial interpolation, some of which are outlined below:

a. Polynomials of degree N do not always pass through every data point (if there are more than N + 1 data points). This is objectionable when the known data points are such

that data points can be considered "exact" (i.e., data points whose probable error of position is small and the polynomial fit does not meet this error crition).

b. Oscillations, as many as $(N - 1)/2$, may occur between the first and last data points. Such oscillations may lie outside the range of the known behavior of the system being modeled or may incorrectly model derivative behavior.

c. It is extremely difficult to predict the overall behavior of an N^{th} degree polynomial when a data point is added, deleted, or "adjusted" (again assuming there are more than N data points). This difficulty transforms the curve-fitting process from a science to an art.

d. Although derivatives of polynomials are continuous, one can not assume that they are representative of the physical situation being modeled.

13. Summary. Polynomial interpolation methods can be successfully applied to many nonlinear problems, but a high level of education, skill, and experience is required of the curve-fitting practitioner (the equation maker). Secondly, polynomials can never be completely trusted. New sets of data, even though they are from the same application, must be validated (plotted and examined), and most likely adjusted through a trial and error process, before the new polynomial fit can be used.

Verification of Interpolation Systems

14. It is important to note that derivative faults are easily overlooked, especially when the user restricts his evaluation of the interpolation procedure to testing its ability to reproduce a physical effect within specified error bounds. As we have demonstrated, one can have a perfectly satisfactory model of the observed behavior, while that same model is invalid in other important physical characteristics. It is, therefore, highly desirable to examine all characteristics of the physical system which the interpolating model is expected to reproduce.

PART II: CUBIC SPLINE INTERPOLATION

15. The aforementioned reasons provide motivation to look for a better, easy-to-use interpolating technique that will give smooth, predictable behavior and have continuous first derivatives. U. S. Army Engineer Waterways Experiment Station (WES) activity in several engineering areas indicates that the cubic spline[2,3] (hereinafter referred to as spline) has many advantages as an interpolating method. Several important spline characteristics are outlined below:

a. The spline passes through the data points.

b. The spline is a piece-wise cubic. The data points mark the points of transition from one set of coefficients to the next.

c. The first and second derivatives of the spline are continuous at all points.

d. The spline curve looks similar to that drawn by a french curve or a mechanical spline (more on mechanical or physical splines in paragraphs 20 and 33).

e. Adjustment of any data point affects the entire curve, but the effect is predictable.

f. The spline is uniquely defined by the X and Y coordinates of each data point and either the first or second derivative at each data point.

Mathematical Formulation

16. The mathematical spline function is a piece-wise third degree (cubic) polynomial passing through all data points and having continuous first and second derivatives.

17. The spline can be viewed as a set of cubic equations, one equation for each interval between successive data points. The coefficients of the cubic equations are such that, at any data point, the equation for the left interval will yield the same values for the first and second derivatives, respectively, as will the equation for the right interval.

18. Given a set of N data points for which the coordinates (X_i, Y_i) and second derivatives (M_i) are known for every point (i = 1, 2, 3...N),

then the interpolating spline function $S(x)$ is defined as

$$S(x) = \frac{1}{6} M_i \frac{(X_{i+1} - x)^3}{X_{i+1} - X_i} + \frac{1}{6} M_{i+1} \frac{(x - X_i)^3}{X_{i+1} - X_i} + \left[Y_i - \frac{1}{6} M_i (X_{i+1} - X_i)^2\right] \frac{X_{i+1} - x}{X_{i+1} - X_i} + \left[Y_{i+1} - \frac{1}{6} M_{i+1} (X_{i+1} - X_i)^2\right] \frac{x - X_i}{X_{i+1} - X_i} \quad (1)$$

where i is such that $X_i \leq x \leq X_{i+1}$. The first and second derivatives are

$$S'(x) = -\frac{M_i}{2} \frac{(X_{i+1} - x)^2}{X_{i+1} - X_i} + \frac{M_{i+1}}{2} \frac{(x - X_i)^2}{X_{i+1} - X_i} + \frac{Y_{i+1} - Y_i}{X_{i+1} - X_i} + \frac{1}{6} (M_i - M_{i+1})(X_{i+1} - X_i) \quad (2)$$

$$S''(x) = M_i \left(\frac{X_{i+1} - x}{X_{i+1} - X_i}\right) + M_{i+1} \left(\frac{x - X_i}{X_{i+1} - X_i}\right) \quad (3)$$

19. The method of evaluating the M_i coefficients is presented in Appendix A, along with a derivation of the above expressions. A compact fitting and interpolating FORTRAN program is included in Appendix B.

Discussion of Spline Characteristics

20. The spline properties discussed in paragraph 15 illuminate the advantages and disadvantages of splines. Properties a, b, c, and d combine to produce an accurate curve having continuous first and second derivatives. This, as mentioned earlier, is very useful in engineering analysis problems. Properties d and e provide a physically based insight for adjusting data to obtain a better curve. In fact, the mathematical cubic spline can be derived from the theory associated with the deflected shape of a weightless elastic beam constrained at particular points.

21. Greville[2] noted that many physical systems are correctly modeled by cubic or quadratic equations. This fact accounts, to a large extent, for the popularity of cubic splines. It also enables us to obtain a

satisfactory fit with a small number of data points when an interval of the physical phenomenon exhibits first, second, or third degree behavior.

Spline versus linear interpolation

22. Fig. 2 is a spline fit to the same set of data used in fig. 1. Note the smoothness of the curve between data points. More important is the continuous derivative behavior. This is a marked improvement over the linear interpolation model shown in fig. 1.

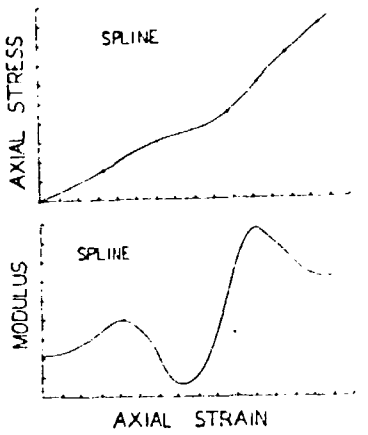


Fig. 2. Spline approximation of stress-strain relation shown fig. 1

Spline versus polynomial interpolation

23. An advantage of splines over polynomials is illustrated in fig. 3 which shows a set of 6 data points to which a fifth degree polynomial and a spline have been fit. This example shows the influence of a "bad" data value on both the spline and polynomial. Note how the spline tends to minimize the influence of the bad point P. The polynomial on the other hand is completely upset by the bad point.

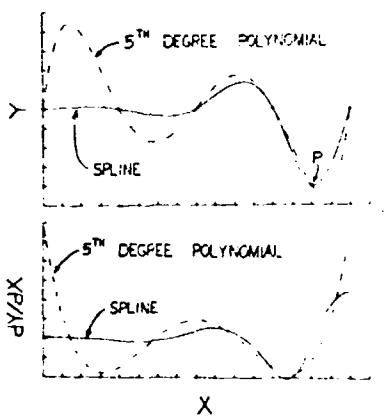


Fig. 3. General comparison of spline and polynomial fits

24. This rather extreme example is introduced not as a practical consideration but rather to illustrate the oscillating character of polynomials that is so troublesome in curve fitting. Discussion of least square polynomials is provided in the applications section, paragraphs 40-49.

Disadvantages of spline

25. Properties b and f (paragraph 15) are sometimes interpreted as disadvantages. The spline is not defined as a single expression over the entire range of the data points. Therefore, to evaluate $S(x)$ one must first find the two adjacent data points bounding x and then apply the equations of paragraph 18. This results in longer computer time than that required in polynomial fits. However, search time can be minimized by use of efficient algorithms.

26. The number of coefficients needed to define the spline (three times the number of data points) is too large for some engineering applications. This is due to the fact that the computer codes themselves (such as the finite element and finite difference codes) require vast amounts of storage to solve physical problems. For instance, the finite element analysis of soil-structure interaction problems often requires that several soil zones be considered. If the nonlinear stress-strain behavior of the soils is modeled by splines, then for each different layer of soil one must store parameters for a different spline function. Thus, the number of storage locations required by splines will be greatly increased in such problems. The same is true for finite difference solutions of ground shock problems. The engineer, quite naturally, is inclined to use the available storage for larger physical problems rather than to introduce what he may feel is an unnecessary elegance in the stress-strain interpolation routine.

Summary

27. Although WES experience with splines is limited, it has led to the conclusion that the advantages of the spline (continuous derivatives, smooth curve, dependability, and physical insight) far outweigh the problems of increased computer memory and somewhat longer computation times. Further, through the skill of the numerical analyst and programmer, one can exercise considerable influence in adapting the formulation and code to minimize run time or storage, a point discussed in the applications section.

PART III: APPLICATION OF SPLINES TO ENGINEERING PROBLEMS

28. This section describes the application of cubic splines to several civil engineering problems at WES. Applications of the cubic spline fitting and interpolating techniques to hydraulics problems, soil-structure interaction problems, and seepage problems are presented.

Hydraulics Problems

Rating curves

29. This application in the field of hydraulic engineering deals with modeling rating curves in a flood routing program authored by Mr. E. A. Graves of the Lower Mississippi Valley Division. Mr. Graves and Mr. J. B. Cheek, Jr., of WES are currently involved in the application of spline methods to this problem.

30. Rating curves are used to describe, at stated points (stations) on a river, the flow (discharge) characteristics as a function of the height of the water in the river (stage). The flood routing program produces very satisfactory results, but a great deal of difficulty is experienced in obtaining polynomial fits to the rating curves that are required as input to the program. The following figures and discussion illustrate the improvement brought by the spline interpolation method to the rating curve model.

31. Spline fit to 20 points. Fig. 4 shows 20 data points of a rating curve with a spline fit through those points. Except for the large number of points, the spline representation was considered satisfactory until the derivative shown in fig. 5 was examined. It was characterized by an objectionable lack of smoothness.

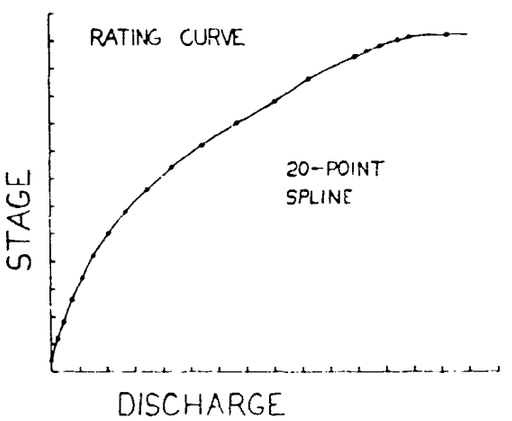


Fig. 4. Spline fit of 20-point rating curve

32. Reducing number of points to improve derivative. In order to

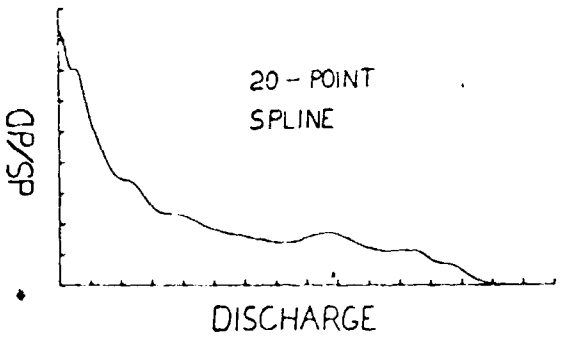


Fig. 5. Derivative for 20-point spline fit

the physical insight to the spline has its impact.

33. As mentioned previously, the mathematical spline formulation also describes the deflection of a weightless linear elastic beam (a physical spline) which is simply constrained at the data points. This means that we can visualize the mathematical spline fit as that shape assumed by a thin strip of steel which is constrained on the paper by a straight pin at each data point. With

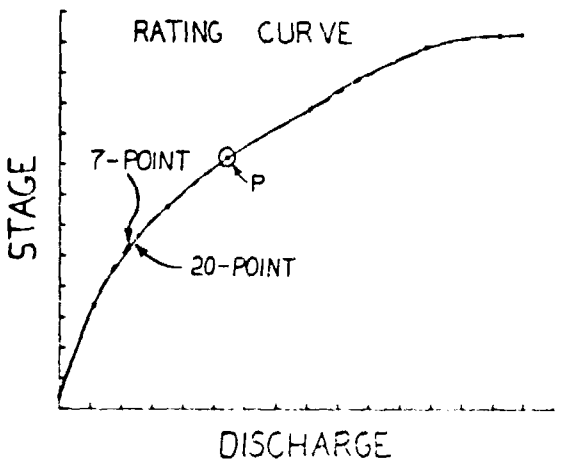


Fig. 7. Comparison of 7- and 20-point spline fits

reduce the number of data points and produce a smoother derivative curve, 6 of 20 original points were chosen and a spline was fitted to them. This is illustrated in fig. 6 along with the original 20-point curve. The 6-point curve does not match the original to the desired degree. It seems desirable to add a few more points. In a situation such as this,

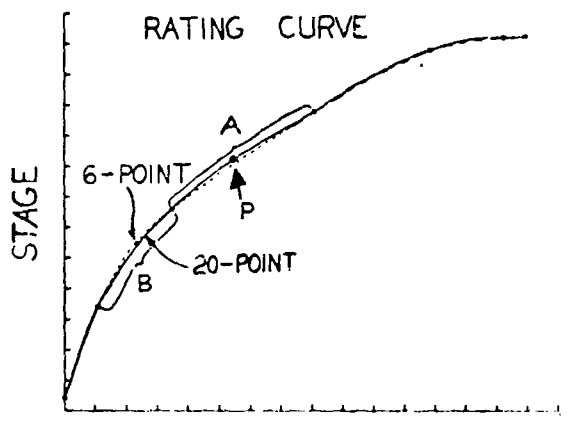


Fig. 6. Comparison of 6- and 20-point spline fits

this concept in mind, the effect of adding point P (fig. 6) to the 6-point data set can be easily predicted. The new point P would bend the spring closer to the original curve along A and would also cause a downward movement along B.

34. Fig. 7, which shows the 7-point spline fit along with the original 20-point fit, shows the effect of adding point P. Note how

nicely the 7-point fit follows the lower portion of the curve. By some further adjustments, the upper portion of the curve could be improved, but it is not necessary for the present purpose since the point has been made, i.e., insight into the physical character of the mathematical spline makes the curve-fitting process easy.

35. The derivative for the 7-point curve (fig. 8) shows much improvement in smoothness. This improvement is presumed to be due to the reduced number of points. This aspect of spline interpolation will be discussed further in paragraphs 43-45.

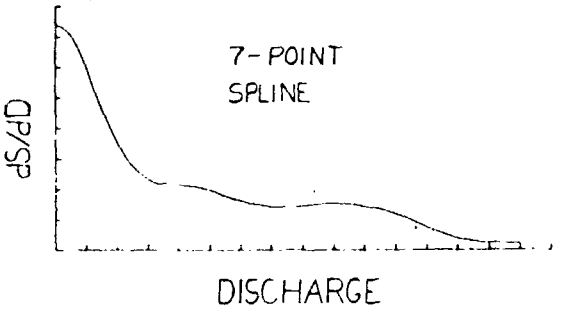


Fig. 8. Derivative for 7-point spline fit

36. Modification of spline fit. Because rating (or conveyance) curves for an alluvial river change with time, it is sometimes necessary to change them. With splines, this can be done by substituting new data points, with the assurance that the new fit will be satisfactory. This contrasts with the use of polynomials, which require many more data points and which must be carefully studied to determine if a satisfactory fit has been obtained. Also, in the application discussed, the first derivative is required, and this can be obtained directly during the spline interpolation procedure. Even in applications where the first derivative is not used, if a polynomial curve fit is to be used, it is desirable to obtain the first derivatives for use in studying the suitability of the curve to the purpose in hand.

Model storage curves

37. The success with modeling rating curves soon led to experimentation with modeling storage curves. This flood routing application has severe computer memory restrictions on it, so extra steps were taken to reduce the coefficients stored. This was accomplished by retaining only the coordinates of the points and beam moment of each data point. This is not normally done because the two other required coefficients for each point, developed during the spline fit process, must be thrown out and recalculated for the two bounding points during the interpolation computation. However, by so doing, it is possible to use the spline in a

PAGES 138, 139
ARE
MISSING
IN
ORIGINAL
DOCUMENT

smoothing parameter. From a practical standpoint, it is generally preferable and less time consuming to select a few data points at the outset rather than to make repeated runs in search of a good parameter for each new set of data. (This is not to imply that one method is superior to the other, rather that interpolating splines can economically be made to function satisfactorily for many scientific-engineering applications.)

44. Fig. 11 shows spline and polynomial fits to 16 of the original 89 points of fig. 9. Note that the points shown are not the result of several trials to obtain the best set. They were simply selected so that the x values were about an inch apart on the original drawing, a procedure which certainly cannot be called "tuning" the data.

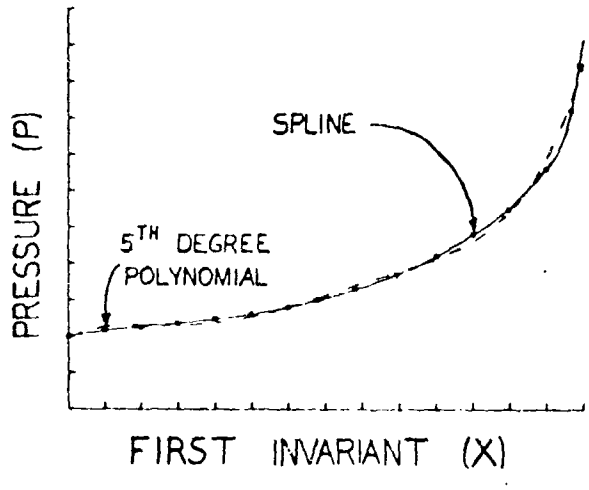


Fig. 11. Comparisons of 16-point spline and polynomial fits

45. Fig. 12 shows the vastly improved spline derivative plot which now appears superior to both the 89- and 16-point polynomial derivatives. Any improvements required can be obtained with the aid of the aforementioned insight about the physical equivalent of the cubic spline (paragraph 33).

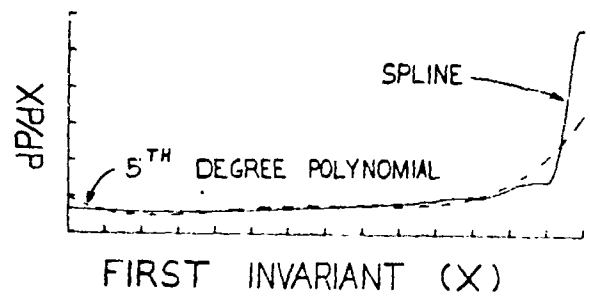


Fig. 12. Comparison of spline and polynomial derivatives for 16-point fits

Summary

46. The authors have been directly involved in three soil-structure interaction projects at WES:

a. R. E. Walker and Cheek (March 1970) on a dynamic nonlinear elastic finite element method stress analysis program.

b. J. L. Kirkland and Cheek (April 1970) on a nonlinear elastic incremental loading soil-structure interaction program.

c. PFC B. Phillips and Radhakrishnan (1970) on a finite difference solution of nonlinear elastic analysis of ground motion.

This experience has indicated that the addition of splines to soil-structure interaction codes is worthwhile. However, it cannot be stated with certainty that the computed results are better, although there are signs of improvement in the stability of the procedure (such as a reduction in strain energy growth), since no known correct solution with which to compare the results exists. It is felt that splines have removed some obvious errors in modeling nonlinear characteristics and have, at the least, allowed us to turn our research to other problems.

Steady-State Seepage Problems

Confined seepage

47. Producing a contour map from a set of data points

$$X_i, Y_i, Z_i \quad ; \quad i = 1, 2, 3 \ldots N$$

has not in general been satisfactorily accomplished. A special case of this problem where the data points occur on a rectangular grid has, however, been solved using splines. Fig. 13 shows the data configuration.

48. To construct a contour map, a surface must first be fitted to the data points. In this section, a method which fits a simple bicubic spline[4] to the data is described, and an application to confined steady-state seepage under a weir is discussed.

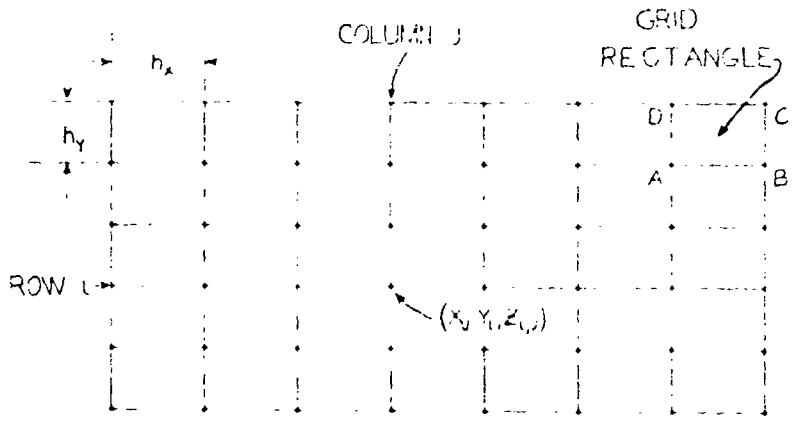


Fig. 13. Data configuration for spline contouring

49. The first part of the procedure is to fit a cubic spline along each row and each column of the data points

$$X_j, Y_i, Z_{i,j} \quad ; \quad i = 1, 2, 3 \ldots i_{max} \quad ; \quad j = 1, 2, 3 \ldots j_{max}$$

where

$$X_j = (j - 1)h_x$$
$$Y_i = (i - 1)h_y$$
$$Z_{i,j} = Z \text{ value corresponding to } (X_j, Y_i)$$

The cubic spline as defined on the j^{th} interval of the i^{th} row from property f of paragraph 15 is

$$S_i^j(x) = \frac{M_{i,j}^{(H)}}{6h_x}(X_{j+1} - x)^3 + \frac{M_{i,j+1}^{(H)}}{6h_x}(x - X_j)^3$$

$$+ \left(Z_{i,j} - \frac{1}{6} M_{i,j}^{(H)} h_x^2\right) \frac{X_{j+1} - x}{h_x} + \left(Z_{i,j+1} - \frac{1}{6} M_{i,j+1}^{(H)} h_x^2\right) \frac{x - X_j}{h_x} \qquad (4)$$

where $M_{i,j}^{(H)}$ is the second derivative of the horizontal spline passing through (i,j). Similarly, the spline as defined on the i^{th} interval of the j^{th} column is

$$S_j^i(y) = \frac{M_{i,j}^{(V)}}{6h_y}(Y_{i+1} - y)^3 + \frac{M_{i+1,j}^{(V)}}{6h_y}(y - Y_i)^3$$

$$+ \left(Z_{i,j} - \frac{1}{6} M_{i,j}^{(V)} h_y^2\right)\left(\frac{Y_{i+1} - y}{h_y}\right) + \left(Z_{i+1,j} - \frac{1}{6} M_{i+1,j}^{(V)} h_y^2\right)\left(\frac{y - Y_i}{h_y}\right) \qquad (5)$$

where $M_{i,j}^{(V)}$ is the second derivative of the vertical spline passing through (i,j).

50. From these cubic splines, the simple bicubic spline is generated. In the region (grid rectangle)

$$X_j \leq x \leq X_{j+1}$$

$$Y_i \leq y \leq Y_{i+1}$$

$$S^{i,j}(x,y) = \frac{1}{2} S^i_j(y) \frac{X_{j+1} - x}{h_x} + \frac{1}{2} S^i_{j+1}(y) \frac{x - X_j}{h_x}$$

$$+ \frac{1}{2} S^j_i(x) \frac{Y_{i+1} - y}{h_y} + \frac{1}{2} S^j_{i+1}(x) \frac{y - Y_i}{h_x} \quad (6)$$

where $S^{i,j}(x,y)$ is the bicubic spline as defined in the grid rectangle whose lower left-hand coordinates are (X_j,Y_i). From this bicubic spline, the actual construction of the contour lines can be accomplished as explained in reference 5.

51. A general purpose contouring program, documented in reference 4, which uses the above method, has been applied to several problems. One of the most important of these is producing equipotential lines from steady-state confined seepage under a weir. The problem is illustrated in fig. 14. It consists of a weir resting on a pervious homogeneous region underlain by an impervious base. Seepage occurs through the pervious medium because of the head differential between its two ends. This seepage produces uplift pressure on the base of the weir, and the engineer is interested in

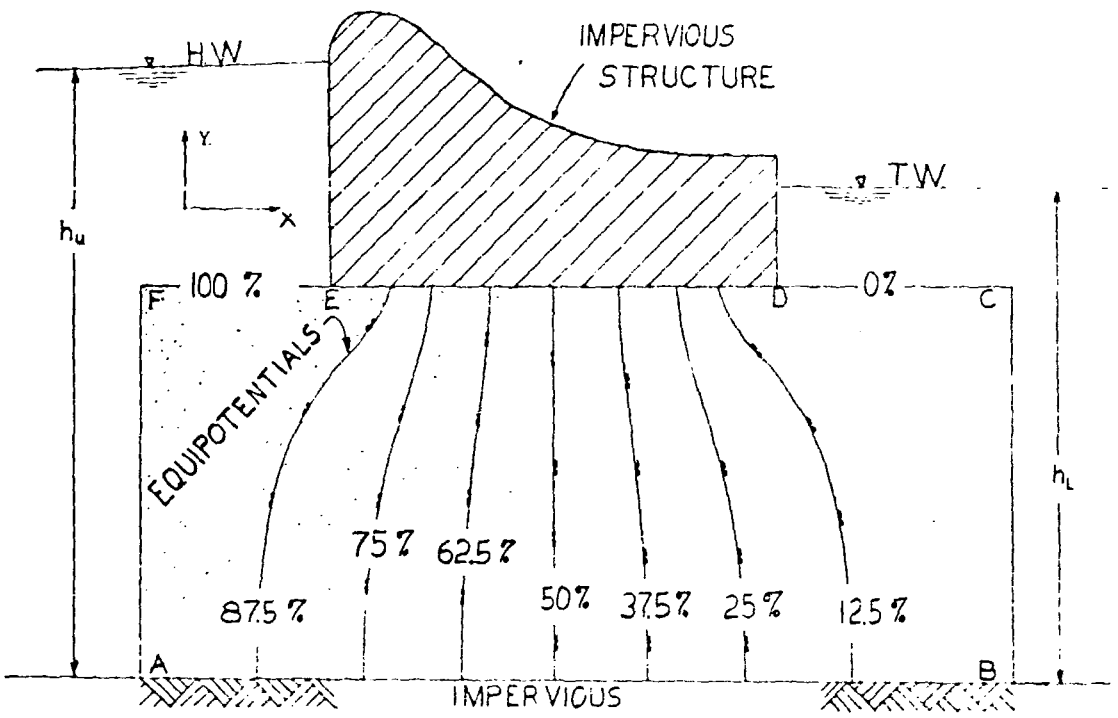


Fig. 14. Steady-state seepage under a weir

the distribution of pressures in the pervious medium. The results are normally expressed as contours of equal pressures or potentials called equipotentials.

52. Region ABCF in fig. 14 is divided into a grid system, and a finite difference solution to the governing partial differential equation (Laplace's equation).

$$\frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2} = 0$$

with boundary conditions

$h = h_U$ on AFE

$h = h_L$ on DCB

$\frac{\partial h}{\partial y} = 0$ on ED and AB (ED and AB are flow lines. There is no flow across these boundaries.)

is obtained.[6] The output from this solution is used as input to the contouring program to obtain the equipotential lines. The results are shown in fig. 14. The equipotentials are labelled as the percentage $\left[(h_e - h_L)/(h_U - h_L)\right] \times 100$, where h_e is a given equipotential.

53. The quality of the contour map can be illustrated in three ways:

a. The contour lines are smooth.

b. The problem was set up such that the center of the weir was in the center of the region. One would therefore expect the 50% equipotential line to be a line of symmetry. This is indeed the case.

c. The boundary condition $\partial h/\partial y = 0$ on ED and AB requires that the equipotential lines intersect the line segments orthogonally. Again, this is, in fact, the situation.

54. It is concluded that the simple bicubic spline is an excellent function for contouring data established on a grid.

Unconfined seepage

55. A problem of classic importance in civil engineering analysis is that of unconfined flow in an earth dam, with special interest given to the phreatic surface,[7,8] which is a top flow line across which there is no flow. Another property of the phreatic surface is that the potential at

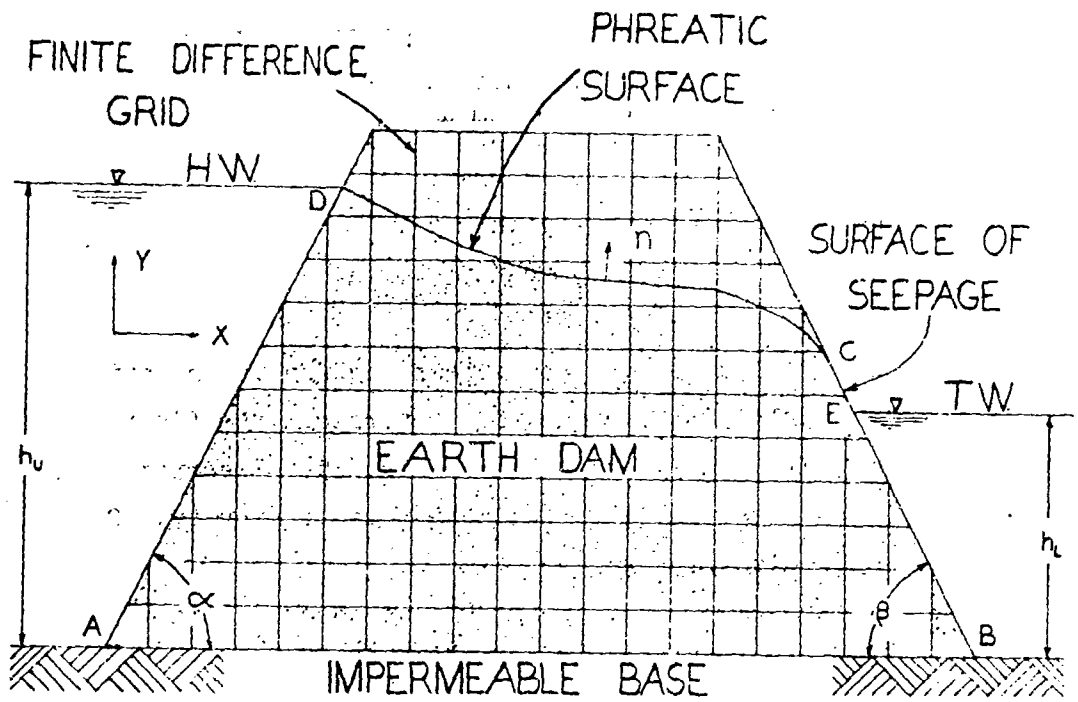


Fig. 15. Location of phreatic surface

any point on the surface is equal to the elevation head at that point.

56. This problem is illustrated in fig. 15. The phreatic surface (which is at atmospheric pressure) is itself derived from the flow equations and must generally be fixed by empirical or trial and error schemes. The trial and error procedure is described here.

57. For steady-state conditions, the governing partial differential equation is

$$\frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2} = 0$$

with boundary conditions

$$\frac{\partial h}{\partial y} = 0 \quad \text{on} \quad AB$$

$$h = h_U \quad \text{on} \quad AD$$

$$h = h_L \quad \text{on} \quad BE$$

$$h = y \quad \text{on} \quad DCE$$

$$\frac{\partial h}{\partial n} = 0 \quad \text{on} \quad DC$$

The procedures used in solving this problem are as follows:

a. Make an initial guess to the phreatic surface DC .

b. Solve the problem by finite difference methods as if it were a confined seepage problem using $\partial h/\partial n = 0$ and $P/\rho g$ = constant on DC (P = pressure ; ρ = mass density of fluid ; g = acceleration due to gravity).

c. Adjust DC to satisfy $h = y$.

d. Repeat b and c until $\partial h/\partial n = 0$ and $h = y$ are satisfied on DC simultaneously.

58. The principal difficulty of this procedure is in satisfying $\partial h/\partial n = 0$ on DC .[8] This is because DC intersects the finite difference grid at irregular points, as illustrated in fig. 15. Incorporating splines into the procedure may essentially eliminate this difficulty. Step b should then be altered as follows:

a. Fit a cubic spline along DC as shown in fig. 15. Note that the points of intersection of the phreatic surface and the grid are used as data points. Use the set of equations which solves for the slopes at the data points (see Appendix A for details), and set

$$S_1 = -\cot \alpha$$

$$S_N = -\tan \beta$$

where N is the number of data points.

b. Replace $\partial h/\partial n = 0$ and $P/\rho g$ = constant on DC with the equivalent expressions

$$\left(\frac{\partial h}{\partial x}\right)_i = \frac{S_i}{1 + S_i^2}$$

$$\left(\frac{\partial h}{\partial y}\right)_i = \frac{S_i^2}{1 + S_i^2}$$

$$i = 1, 2, 3 \ldots N$$

where the S_i's are the slopes as computed by the spline fit.

c. Incorporate these simple formulas into the finite difference solution to obtain the next trial values.

PART IV: CONCLUSIONS

59. The spline interpolation technique is a valuable tool in curve fitting for computer programs for which more commonly used techniques are unsuitable or of limited value. The spline technique has three primary advantages:

a. The spline passes through all data points.

b. The first and second derivatives of the spline are continuous at all points.

c. The spline can be easily modified to satisfy new or additional data.

WES experience in applying spline techniques to engineering problems has indicated that in many applications these advantages outweigh the additional storage and/or computation time requirements of the technique.

60. Since the spline function is required to pass through all data points, erratic derivative behavior may result from experimental error when the data points are numerous and closely spaced. Trial and error methods for smoothing such functions exist, but they are time consuming. WES experience has indicated that acceptable results can generally be obtained by simply selecting a more limited, more widely spaced set of the data points to which to fit the curve.

LITERATURE CITED

1. Zienkiewicz, O. C. and Cheung, Y. K., The Finite Element Method in Structural and Continuum Mechanics, McGraw-Hill, London, 1967.
2. Greville, T. N. E., "Data Fitting by Spline Functions," MRC Technical Summary Report No. 893, June 1968, Mathematics Research Center, U. S. Army, University of Wisconsin, Madison, Wisc.
3. Ahlberg, J. H., Nilson, E. N., and Walsh, J. L., Theory of Splines and Their Applications, Academic Press, 1967.
4. De Boor, C., "Bicubic Spline Interpolation," Journal of Mathematics and Physics, Vol 41, 1962, pp 212-218.
5. Long, J. T. and Tracy, F. T., "A General Purpose Contouring System," Miscellaneous Paper T-70-1, Apr 1970, U. S. Army Engineer Waterways Experiment Station, CE, Vicksburg, Miss.
6. Shaw, F. S., "Second Order Linear Partial Differential Equations," An Introduction to Relaxation Methods, Dover, New York, 1953, pp 71-84.
7. Shaw, F. S. and Southwell, R. V., "Relaxation Methods Applied to Engineering Problems; VII: Problems Relating to the Percolation of Fluids Through Porous Materials," Proceedings, Royal Society, Vol 178, No. 1, 9 May 1971, pp 1-17.
8. Finnemore, E. J. and Perry, B., "Seepage Through an Earth Dam Computed by the Relaxation Technique," Water Resources Research, Vol 4, No. 5, Oct 1968, pp 1059-1067.

APPENDIX A: EQUATIONS FOR CUBIC SPLINE

1. A cubic spline is a function whose third derivative is a step function with points of discontinuity at the data points

$$X_i, Y_i \quad ; \quad i = 1, 2, 3 \ldots N$$

where N is the number of data points. The spline function of degree three is therefore a piece-wise, continuous third degree polynomial having continuous first and second derivatives.

2. For this application the function will be further required to interpolate the data points. Hence, on the i^{th} interval

$$X_i \leq x \leq X_{i+1} \quad ; \quad i = 1, 2, 3 \ldots N - 1$$

$$S(x) = a_i + b_i x + c_i x^2 + d_i x^3 \tag{A1}$$

where a_i , b_i , c_i , and d_i are constants to be evaluated. Equation A1 can be rewritten as

$$S(x) = a_i'(x - X_i)^3 + b_i'(X_{i+1} - x)^3$$

$$+ c_i'(x - X_i) + d_i'(X_{i+1} - x) \tag{A2}$$

where a_i' , b_i' , c_i' , and d_i' are an alternate set of constants. Differentiating equation A2 twice yields

$$S''(x) = 6a_i'(x - X_i) + 6b_i'(X_{i+1} - x) \tag{A3}$$

Applying

$$S''(X_i) = M_i \tag{A4}$$

$$S''(X_{i+1}) = M_{i+1} \tag{A5}$$

where M_i is the second derivative at point (X_i, Y_i), the result is

$$a_i' = \frac{M_{i+1}}{6(X_{i+1} - X_i)} \quad \text{and} \quad b_i' = \frac{M_i}{6(X_{i+1} - X_i)} \tag{A6}$$

3. Since the spline must interpolate the data points

$$S(X_i) = Y_i \tag{A7}$$

$$S(X_{i+1}) = Y_{i+1} \tag{A8}$$

Substituting these into equation A2 produces

$$c_i' = \left[Y_{i+1} - a_i'\left(X_{i+1} - X_i\right)^3\right]\left(\frac{1}{X_{i+1} - X_i}\right)$$

$$= \left[Y_{i+1} - \frac{M_{i+1}}{6}\left(X_{i+1} - X_i\right)^2\right]\left(\frac{1}{X_{i+1} - X_i}\right) \tag{A9}$$

$$d_i' = \left[Y_i - b_i'\left(X_{i+1} - X_i\right)^3\right]\left(\frac{1}{X_{i+1} - X_i}\right)$$

$$= \left[Y_i - \frac{M_i}{6}\left(X_{i+1} - X_i\right)^2\right]\left(\frac{1}{X_{i+1} - X_i}\right) \tag{A10}$$

Collecting terms

$$S(x) = \frac{1}{6} M_1 \frac{\left(X_{i+1} - x\right)^3}{X_{i+1} - X_i} + \frac{1}{6} M_{i+1} \frac{\left(x - X_i\right)^3}{X_{i+1} - X_i}$$

$$+ \left[Y_i - \frac{1}{6} M_i\left(X_{i+1} - X_i\right)^2\right] \frac{X_{i+1} - x}{X_{i+1} - X_i}$$

$$+ \left[Y_{i+1} - \frac{1}{6} M_{i+1}\left(X_{i+1} - X_i\right)^2\right] \frac{\left(x - X_i\right)}{X_{i+1} - X_i} \tag{A11}$$

Differentiating

$$S'(x) = -\frac{M_i}{2}\frac{(X_{i+1} - x)^2}{X_{i+1} - X_i} + \frac{M_{i+1}}{2}\frac{(x - X_i)^2}{X_{i+1} - X_i}$$

$$+ \frac{Y_{i+1} - Y_i}{X_{i+1} - X_i} + \frac{1}{6}(M_i - M_{i+1})(X_{i+1} - X_i) \qquad \text{(A12)}$$

$$S''(x) = M_i\left(\frac{X_{i+1} - x}{X_{i+1} - X_i}\right) + M_{i+1}\left(\frac{x - X_i}{X_{i+1} - X_i}\right) \qquad \text{(A13)}$$

$$X_i \leq x \leq X_{i+1} \quad ; \quad i = 1, 2, 3 \ldots N - 1$$

4. The M_i's are evaluated by satisfying continuity of the first derivative at the data points

$$X_i, Y_i \quad ; \quad i = 2, 3, 4 \ldots N - 1$$

and specifying M_1 and M_N. At point i

$$S'(X_i-) = S'(X_i+)$$

Applying equation A12

$$\frac{M_i}{2}(X_i - X_{i-1}) + \frac{Y_i - Y_{i-1}}{X_i - X_{i-1}} + \frac{1}{6}(M_{i-1} - M_i)(X_i - X_{i-1})$$

$$= -\frac{M_i}{2}(X_{i+1} - X_i) + \frac{Y_{i+1} - Y_i}{X_{i+1} - X_i} + \frac{1}{6}(M_i - M_{i+1})(X_{i+1} - X_i) \qquad \text{(A14)}$$

Collecting terms

$$M_{i-1}(X_i - X_{i-1}) + 2M_i(X_{i+1} - X_{i-1}) + M_{i+1}(X_{i+1} - X_i)$$
$$= 6\left(\frac{Y_{i+1} - Y_i}{X_{i+1} - X_i} + \frac{Y_i - Y_{i-1}}{X_i - X_{i-1}}\right) \quad \text{(A15)}$$

$$i = 2, 3, 4 \ldots N - 1$$

A good choice for M_1 and M_N is

$$M_1 = M_N = 0 \quad \text{(A16)}$$

This set of simultaneous linear equations can be solved for the M_i's .

5. It is sometimes more desirable to solve for first derivatives at the data points rather than second derivatives. This set of simultaneous equations is derived as follows. Satisfying

$$S'(X_i) = T_i \quad \text{(A17)}$$

$$S'(X_{i+1}) = T_{i+1} \quad \text{(A18)}$$

where T_i is the first derivative at (X_i, Y_i), equation A12 becomes

$$T_i = -\frac{M_i}{2}(X_{i+1} - X_i) + \frac{Y_{i+1} - Y_i}{X_{i+1} - X_i} + \frac{1}{6}(M_i - M_{i+1})(X_{i+1} - X_i) \quad \text{(A19)}$$

$$T_{i+1} = \frac{M_{i+1}}{2}(X_{i+1} - X_i) + \frac{Y_{i+1} - Y_i}{X_{i+1} - X_i} + \frac{1}{6}(M_i - M_{i+1})(X_{i+1} - X_i) \quad \text{(A20)}$$

Collecting terms

$$T_i = \left(-\frac{M_i}{3} - \frac{M_{i+1}}{6}\right)(X_{i+1} - X_i) + \frac{Y_{i+1} - Y_i}{X_{i+1} - X_i} \quad \text{(A21)}$$

$$T_{i+1} = \left(\frac{M_i}{6} + \frac{M_{i+1}}{3}\right)(X_{i+1} - X_i) + \frac{Y_{i+1} - Y_i}{X_{i+1} - X_i} \quad \text{(A22)}$$

Solving for M_i and M_{i+1}

$$M_i = \left(3\frac{Y_{i+1} - Y_i}{X_{i+1} - X_i} - 2T_i - T_{i+1}\right)\left(\frac{2}{X_{i+1} - X_i}\right) \tag{A23}$$

$$M_{i+1} = \left(T_i + 2T_{i+1} - 3\frac{Y_{i+1} - Y_i}{X_{i+1} - X_i}\right)\left(\frac{2}{X_{i+1} - X_i}\right) \tag{A24}$$

Requiring continuity of the second derivative at data points $i = 2, 3, 4 \ldots N - 1$

$$S''(X_i+) = S''(X_i-) \tag{A25}$$

Applying equations A12 and A24

$$\left(3\frac{Y_{i+1} - Y_i}{X_{i+1} - X_i} - 2T_i - T_{i+1}\right)\left(\frac{2}{X_{i+1} - X_i}\right) = \left(T_{i-1} + 2T_i - 3\frac{Y_i - Y_{i-1}}{X_i - X_{i-1}}\right)\left(\frac{2}{X_i - X_{i-1}}\right) \tag{A26}$$

Or

$$(X_{i+1} - X_i)T_{i-1} + 2(X_{i+1} - X_{i-1})T_i + (X_i - X_{i-1})T_{i+1} = 3\left[\left(\frac{X_{i+1} - X_i}{X_i - X_{i-1}}\right)(Y_i - Y_{i-1}) + \left(\frac{x - X_{i-1}}{X_{i+1} - X_i}\right)(Y_{i+1} - Y_i)\right] \tag{A27}$$

$$i = 2, 3, 4 \ldots N - 1$$

$M_1 = M_N = 0$ becomes

$$2T_1 + T_2 = 3\frac{Y_2 - Y_1}{X_2 - X_1} \tag{A28}$$

$$T_{N-1} + 2T_N = 3\frac{Y_N - Y_{N-1}}{X_N - X_{N-1}} \tag{A29}$$

This system may be solved for the first derivatives at the data points.

APPENDIX B: SPLINE FITTING AND INTERPOLATING SUBROUTINES

1. This appendix contains listings of the two FORTRAN language subroutines, SPLINE (table B1[2]) and SPLINT (table B2). The following sections describe the use of the two routines.

Subroutine SPLINE

2. The SPLINE subroutine is used to fit a cubic spline to a set of (x,y) data. That is, it calculates the moments at each interior data point, assigning zero to the moments at the end points. Note that only three arrays are required for each spline, an x , a y , and a moment array. The spline fit is accomplished by a call statement in the user's program of the form:

CALL SPLINE (A1, A2, N3, A4)

where the arguments A1, A2, N3, and A4 are as follows:

Argument	Purpose
A1	Names the independent variable array (x).
A2	Names the dependent variable array (y).
N3	Specifies the number of (x,y) points. N3 must be in integer form.
A4	Names the array into which SPLINE will store the calculated moments.

3. Note that the user must specify the size of arrays A1, A2, and A4 through DIMENSION or COMMON statements. Several splines may be fit and saved for subsequent use by making successive calls to SPLINE using different names for arguments A1, A2, and A4 and indicating the number of points through argument N3.

4. Example: Given two sets of (x,y) data, fit a spline to each set of data.

5. Solution: Let one set of data be in arrays X1 and Y1 having N1 points. Let the other set of data be in X2, Y2 having N2 points. The following FORTRAN statements are required, assuming that there are no more than 30 points in either data set.

```
DIMENSION X1(30), Y1(30), C1(30), X2(30), Y2(30), C2(30)
                          :
          Statements to input the two data
          sets and specify their size in
          N1 and N2.  (Note that C1 and C2
          need not be set to any specific
          value.)
                          :
CALL SPLINE (X1, Y1, N1, C1)
CALL SPLINE (X2, Y2, N2, C2)
```

6. At this point array C1 will contain N1 moments, and array C2 will contain N2 moments; the first and last moment in each array will be zero.

7. Note that N1 need not equal N2 but neither may be less than 2 nor greater than the maximum size specified for the associated arrays (30 in this example).

Subroutine SPLINT

8. Subroutine SPLINT (SPLine INTerpolate) operates on a cubic spline defined by the (x,y) coordinates of N points and the moment at each of those points. It calculates values of y and y' for any value XX of the independent variable x. Should the value XX lie beyond the range of data defined by the points, this program will extrapolate linearly from the first (or last) using the slope of the spline at that end point.

9. Interpolating is accomplished by a call statement of the form:

```
CALL SPLINT (A1, A2, A3, A4, A5, A6, N7)
```

where the arguments A1-A6 and N7 are as follows:

Argument	Purpose
A1	Specifies the value XX of the independent variable x for which y and y' are desired.
A2	Receives the value of y at XX computed by SPLINT.
A3	Receives the value of y' at XX computed by SPLINT.
A4	Names the independent variable array of the spline.
A5	Names the dependent variable array of the spline.
A6	Names the moment array of the spline.
N7	Specifies the number of (x,y) points that define the spline. N7 must be in integer form.

10. Note that arrays A4, A5, and A6 must contain N7 values each of x , y , and moment, respectively; i.e., they contain the spline defining data. Argument A1 is an input argument for XX , while A2 and A3 receive the values for y and y' calculated by SPLINT.

11. Two of the variables in subroutine SPLINT may be useful in some applications. Variable FPPXX contains the value for y". Variable M indicates whether the computation was an interpolation, in which case M is zero, or an extrapolation, in which case M is -1. A reduction in run time would likely result from incorporating a more sophisticated search procedure than that used (statement numbers 100 through 140 in table B2).

12. Example: Assuming that the steps outlined in the example for subroutine SPLINE have been taken, the following statements would calculate y values (YY) and y' (YPRIME) at XX = 36.49 from the second set of spline data.

```
XX = 36.49

CALL SPLINT (XX, YY, YPRIME, X2, Y2, C2, N2)
```

Test Program

13. Table B3 shows a simple test program and the ten interpolated values computed using the GE 430 Time Sharing system at the U. S. Army Engineer Waterways Experiment Station.

Table B1

Subroutine SPLINE

```
1000        SUBROUTINE SPLINE (X, ZY, N, S2)
1010C       SPLINE FITTING SUBROUTINE ADAPTED FROM
1020C       WORK BY GREVILLE, U S ARMY MATH. RESEARCH CENT.
1030C       UNIV OF WISCONSON, T S REPORT \893,
1040C       JUNE 1968.
1050        DIMENSION X(1), ZY(1), S2(1)
1060       DATA EPSLN /1.E-6/
1070        N1 = N - 1
1080        ASSIGN 110 TO ISW
1090        DO 130 I = 1, N1
1100        H = X(I + 1) - X(I)
1110        DLY = (ZY(I + 1) - ZY(I)) / H
1120        GO TO ISW, (110, 100)
1130  100   H2ZZZ = HL + H
1140        S2(I) = 2. * (DLY - YL) / H2ZZZ
1150        GO TO 120
1160  110   ASSIGN 100 TO ISW
1170  120   HL = H
1180        YL = DLY
1190  130   CONTINUE
1200C
1210        S2(1) = 0.
1220        S2(N) = 0.
1230        OMEGA = - 1.0717968
1240  140   ETA = 0.
1250        ASSIGN 170 TO ISW1
1260        DO 190 I = 1, N1
1270        H = X(I + 1) - X(I)
1280        DLY = (ZY(I + 1) - ZY(I)) / H
1290        GO TO ISW1, (170, 150)
1300  150   H2ZZZ = HL + H
1310        BI = .5 * HL / H2ZZZ
1320        W = (BI * S2(I - 1) + (.5 - BI) * S2(I + 1) + S2(I) +
1330%        3. * (YL - DLY) / H2ZZZ) * OMEGA
1340        S2(I) = S2(I) + W
1350        Z = ABS(W)
1360        IF (Z - ETA) 180, 180, 160
1370  160   ETA = Z
1380        BETA = S2(I) - W
1390        GO TO 180
1400  170   ASSIGN 150 TO ISW1
1410  180   HL = H
1420        YL = DLY
1430  190   CONTINUE
1440        IF (ABS(BETA) * EPSLN - ETA) 140, 140, 200
1450  200   CONTINUE
1460        RETURN
1470        END
```

Table B2

Subroutine SPLINT

```
1520        SUBROUTINE SPLINT (XB, FXX, FPXX, X, ZY, S2, N)
153CC       SPLINE INTERPOLATING SUBROUTINE , BY JAY CHEEK
1540        DIMENSION X(1), ZY(1), S2(1)
1550        MM = 0
1560        XP = XB
1570        I = 1
1580        IF (XP - X(1)) 100, 170, 110
1590  100   MM = - 1
1600        XP = X(1)
1610        GO TO 170
1620  110   IF (XP - X(N)) 130, 150, 140
1630  120   IF (XP - X(I)) 160, 170, 130
1640  130   I = I + 1
1650        GO TO 120
1660  140   MM = - 1
1670        XP = X(N)
1680  150   I = N
1690  160   I = I - 1
1700  170   HT1 = XP - X(I)
1710        HT2 = XP - X(I + 1)
1720        PROD = HT1 * HT2
1730        DX = X(I + 1) - X(I)
1740        DELY = (ZY(I + 1) - ZY(I)) / DX
1750        S3 = (S2(I + 1) - S2(I)) / DX
1760        FPPXX = S2(I) + HT1 * S3
1770        DELSQS = (S2(I) + S2(I + 1) + FPPXX) / 6.
1780        FXX = ZY(I) + HT1 * DELY + PROD * DELSQS
1790        FPXX = DELY + (HT1 + HT2) * DELSQS + PROD * S3 / 6.
1800        IF (MM.EQ.0) GO TO 180
1810        FXX = FXX + FPXX * (XB - XP)
1820  180   CONTINUE
1830        RETURN
1840        END

READY
```

Table B3

Test Program for SPLINE and SPLINT Subroutines

```
1010      DIMENSION X(10), Y(10), C(10)
1020C   SET THE TEST DATA.
 1030      X(1) = 1.6
1040      Y(1) = 1.
1050      X(2) = 5.4
1060      Y(2) = 2.
1070      X(3) = 7.
1080      Y(3) = 1.
1090      X(4) = 8.2
1100      Y(4) = 1.
1110      NUMB = 4
1120C
1130C   FIT A SPLINE THROUGH THE X,Y DATA POINTS.
1140      CALL SPLINE (X, Y, NUMB, C)
1150C
1160C   INTERPOLATE AT INTERVALS OF ONE.
1170      PRINT 300
1180      DO 100 I=1, 10
1190      XI = I
1200      CALL SPLINT (XI, YY, YP, X, Y, C, NUMB)
1210  100 PRINT 200, XI, YY, YP
 1220      STOP
1230  200 FORMAT (3X 3E20.9)
1240  300 FORMAT (//10X 1HX, 19X 1HY, 19X 7HY PRIME /)
1250      END

READY
RUN

.1.1.1        08:31   WES    05/19/71


          X                      Y                      Y PRIME

      0.100000000E+01      0.606953327E+00      0.655077788E+00
      0.200000000E+01      0.126029407E+01      0.642049980E+00
      0.300000000E+01      0.184263327E+01      0.495487139E+00
      0.400000000E+01      0.219698582E+01      0.186076697E+00
      0.500000000E+01      0.216050413E+01      -.286181346E+00
      0.600000000E+01      0.160917168E+01      -.727131570E+00
      0.700000000E+01      0.100000000E+01      -.338579530E+00
      0.800000000E+01      0.967082546E+00      0.155182285E+00
      0.900000000E+01      0.113543181E+01      0.169289765E+00
      0.100000000E+02      0.130472158E+01      0.169289765E+00
STOP

RUNNING TIME:     3.2 SECS    I/O TIME :      .4 SECS
```

A CORRECTING PROCESS DESIGNED TO CONTROL PROPAGATED ERROR

Isaac S. Metts, Jr.
Walter Reed Army Institute of Research
Walter Reed Army Medical Center
Washington, D. C.

Abstract. A correcting process which allows the differential equation being considered to choose the corrector to be used at each step is formed by controlling the propagation coefficients. It is investigated for stability and effect on propagated error and numerical results are given to support its validity.

1. Introduction. In this paper we shall introduce a new family of corrector formulas and a new correcting process specifically designed to control the growth of propagated error. We will restrict our attention to the solving of differential equations of the form $y' = f(x,y)$, $y_o = y(x_o)$. This paper is related to a paper (P-1173) submitted to "Mathematics of Computation" by Professor James R. Wesson of Vanderbilt University in early 1967.

In part two of the paper we will present background information in the same format as followed by many papers in this area. One such paper which provided particularly helpful guidance was presented by T. E. Hall and A. C. R. Newberry [5].

2. Background Information. We a e concerned with correctors which are members of the family of "closed" formulas of the form

$$(1) \qquad y_{n+1} - A_n y_n - A_{n-1} y_{n-1} - \dots - A_{n-k} y_{n-k} =$$

$$h[B_{n+1}\, y'_{n+1} + B_n y'_n + \dots + B_{n-k}\, y'_{b-k}] + R_{n+1} ,$$

where R_{n+1} is the truncation error for the $(n+1)^{st}$ application of the corrector and h is the step size for equally spaced data.

To insure uniform lines of thought, we give several definitions. Associated with (1) are two defining polynomials

$$(2) \qquad \rho(s) = s^{k+1} - A_n s^k - A_{n-1} s^{k-1} - \dots - A_{n-k}$$

and

(3) $$\sigma(r) = B_{n+1}r^{k+1} + B_n r^k + \ldots + B_{n-k} .$$

Definition 1. For a formula to be Dahlquist stable or Asymptotically stable requires

(a) all roots of (2) be within or on the unit circle, and

(b) those roots lying on the unit circle must have multiplicity one [2].

It should be noted that this is not necessarily the condition of stability which leads to the most favorable propagation of error. This is illustrated by the fact that the well-known corrector, Simpson's Rule, is Dahlquist stable but propagates error most unfavorably. We are therefore more interested in two other variations of this idea of stability which concern themselves with the roots of

(4) $$\tau(r) = (1-KB_{n+1})r^{k+1} - (A_n+KB_n)r^k - \ldots - (A_{n-k} + KB_{n-k}),$$

where $K = h\frac{\partial f}{\partial y}$ (we will assume this partial always exists). We see from [5] that if restrictions are placed on the corrector coefficients to assure that the formula is of reasonably high degree, one root of (4), call it r_1 (also called the principal root), is approximated by e^K. The other k roots are extraneous roots which have been introduced by approximating the roots of a first order differential equation by those of a $(k+1)^{st}$ order difference equation.

Definition 2. A corrector formula is said to be relatively stable if if $|r_i| < |r_1|$ for $i = 2,3,\ldots, k+1$ [4, 7, 8].

Definition 3. A corrector formula is said to be absolutely stable if $|r_i| < 1$ for $i = 2, 3, \ldots, k+1$ [4, 7].

By the n^{th} propagated error of a formula we mean the difference between the n^{th} computed value of y_n of a differential equation and the n^{th} exact value Z_n and denote it by $Z_n - y_n = \varepsilon_n$. Using this notation for propagated error, R for truncation error, and $K = h\frac{\partial f}{\partial y}$, we state the propagation equation for (1) as

$$(5)\qquad \varepsilon_{n+1} - \frac{(A_n + KB_n)}{(1 - KB_{n+1})}\,\varepsilon_n - \frac{(A_{n-1} + KB_{n-1})}{(1 - KB_{n+1})}\,\varepsilon_{n-1} - \cdots - \frac{(A_{n-k} + KB_{n-k})}{(1 - KB_{n+1})}\,\varepsilon_{n-k} = \frac{1}{1 - KB_{n+1}}\,R.$$

3. Preliminary Considerations. If we consider the subfamily of (1) with order 5 and degree 4, we obtain

$$(6)\qquad y_{n+1} = A_3y_n + A_2y_{n-1} + A_1y_{n-2} + A_0y_{n-3} + h[B_4y'_{n+1} + B_3y'_n + B_2y'_{n-1} + B_1y'_{n-2} + B_0y'_{n-3}] + R_{n+1},$$

where the coefficients may be written in the following parametric form

$$A_0 = 1 - A_3 - A_2 - A_1 ,$$
$$A_1 = A_1 ,$$
$$A_2 = A_2 ,$$
$$A_3 = A_3 ,$$
$$B_0 = \frac{1}{720}(224 - 243A_3 - 232A_2 - 251A_1) ,$$
$$B_1 = \frac{1}{360}(512 - 459A_3 - 496A_2 - 323A_1) ,$$
$$B_2 = \frac{1}{240}(128 - 216A_3 - 64A_2 + 88A_1) ,$$
$$B_3 = \frac{1}{360}(512 - 189A_3 - 16A_2 - 53A_1) ,$$
$$B_4 = \frac{1}{1440}(448 + 54A_3 + 16A_2 + 38A_1) .$$

If the error term for (6) can be written exactly, it is of the form

$$(7)\qquad R_{n+1} = E_{n+1}h^6y^{(6)}(\theta)$$

where $\theta\varepsilon[x_0, x_{n+1}]$ and

$$(8)\qquad E_{n+1} = \frac{-27A_3 - 16A_2 - 27A_1}{720} .$$

The propagation equation of (6) is

$$\varepsilon_{n+1} = \alpha\varepsilon_n + \beta\varepsilon_{n-1} + \gamma\varepsilon_{n-2} + \delta\varepsilon_{n-3} + \frac{R_{n+1}}{(-54KA_3 - 16KA_2 - 38KA_1 - 448A + 1440)}, \tag{9}$$

where

$$\alpha(K,A_1,A_2,A_3) = \frac{A_3(1440 - 756K) - 64KA_2 - 212KA_1 + 2048K}{-54KA_3 - 16KA_2 - 38KA_1 - 448K + 1440}, \tag{10}$$

$$\beta(K,A_1,A_2,A_3) = \frac{-1296KA_3 + (1440 - 384K)A_2 + 528KA_1 + 768K,}{-54KA_3 - 16KA_2 - 38KA_1 - 448K + 1440},$$

$$\gamma(K,A_1,A_2,A_3) = \frac{-1836KA_3 - 1984KA_2 + (1440 - 1292K)A_1 + 2048K}{-54KA_3 - 16KA_2 - 38KA_1 - 448K + 1440}$$

$$\delta(K,A_1,A_2,A_3) = \frac{(-1440 - 486KA_3 + (-1440 - 464K)A_2 + (-1440 - 502K)A_1}{-54KA_3 - 16KA_2 - 38KA_1 - 448K + 1440} + \frac{1440 + 448K}{-54KA_3 - 16KA_2 - 38KA_1 - 448K + 1440}$$

For the remainder of this paper, the restriction $|K| < .4$, which is supported in the literature [3, p. 198], will be assumed.

4. Development of the Correcting Process. In developing our subfamily of (6), we are interested primarily in minimizing, or at least controlling, growth of propagated error while at the same time insuring stability. Since in the classical approach propagated error is considered to be controlled by the roots of the characteristic equation of the propagation equation, we have decided to try to control these roots by controlling the propagation coefficients (10). In this paper we have required that they be equal at each step. An extension we hope to make in a later paper is to make a different but theoretically feasible

restriction on the propagation coefficients and compare the two procedures.

Theorem 1. If we require

$$\alpha(K,A_1,A_2,A_3) = \beta(K,A_1,A_2,A_3) = \gamma(K,A_1,A_2,A_3) = \delta(K,A_1,A_2,A_3),$$

Then α, β, γ, δ, A_1, A_2, A_3 all become functions of K and

$$(11) \qquad \alpha(K) = \beta(K) = \gamma(K) = \delta(K) = \frac{12K^4 + 50K^3 + 105K^2 + 120K + 60}{8K^4 - 50K^3 + 120K^2 - 120K + 240},$$

$$A_1(K) = \frac{-56K^3 - 20K^2 - 10K + 72}{-20K^3 + 121K^2 - 49K + 288},$$

$$A_2(K) = \frac{72K^3 - 24K^2 + 48K + 72}{-20K^3 + 121K^2 - 49K + 288},$$

$$A_3(K) = \frac{-72K^3 + 84K^2 - 214K + 72}{-20K^3 + 121K^2 - 49K + 288}.$$

Proof: The proof consists of equating the propagation coefficients pairwise and solving the resulting system of linear equations.

Corollary 1.1. If $|K| < .4$, then $\alpha(K)$ is a continuous, increasing function of K and $\frac{1}{39} < \alpha(K) < \frac{17}{25}$.

From these results we see that we can control the corrector being used by monitoring $K = h\frac{\partial f}{\partial y}$.

5. Results on Stability and Error Propagation.

Theorem 2. If the propagation coefficients (10) are required to be equal and $|K| < .4$, then the correctors of the resulting subfamily satisfy

(1) conditions for absolute stability,

(2) conditions for relative stability,

(3) conditions for Dahlquist stability.

Proof: Since conditions (1) and (2) are concerned with roots of the characteristic equation of (9) with $\alpha=\beta=\gamma=\delta$

$$\tau(r) = r^4 - \alpha(K)r^3 - \alpha(K)r^2 - \alpha(K)r - \alpha(K), \tag{12}$$

we combine our investigations of these conditions. We first find the resolvent cubic of (12)

$$x^3 + \frac{\alpha(K)}{2}x^2 + \frac{1}{4}[\alpha^2(K) + 4\alpha(K)]x + \frac{1}{8}[3\alpha^2(K) + \alpha^3(K)].$$

Using this we find that the discriminant of (12) is

$$D = -16\alpha^6(K) - 88\alpha^5(K) - 203\alpha^4(K) - 256\alpha^3(K).$$

Since $\alpha(K) > 0$, we have $D < 0$ and therefore (12) has two real and two conjugate imaginary roots. We call r_1 the principal real root and r_3, r_4 the conjugate imaginary roots. These roots satisfy

$$-\alpha(K) = r_1 \cdot r_2 \cdot r_3 \cdot r_4$$

and

$$|r_3| = |r_4| = \sqrt{\frac{-\alpha(K)}{r_1 \cdot r_2}}.$$

With this background, our proof of absolute and relative stability is achieved by refining inequalities sufficiently to locate the above roots.

In satisfying condition (3), we are concerned with the roots of

$$\rho(r) = r^4 - A_3r^3 - A_2r^2 - A_1r - A_0. \tag{13}$$

Since $1 - A_3 - A_2 - A_1 - A_0 = 0$, we have the principal root of (13) is $r_1 = 1$. Factoring we obtain

$$\rho^*(r) = r^3 + (1 - A_3)r^2 + (1 - A_3 - A_2)r + A_0. \tag{14}$$

It is not difficult to show that (14) has one negative real root r_2 and two conjugate imaginary roots r_3, r_4. Also $r_2 \cdot r_3 \cdot r_4 = -A_0$ and $|r_3| = |r_4| = \sqrt{\frac{-A_0}{r_2}}$. We again refine inequalities to locate the roots.

With this result, we have shown that the correctors used throughout our process will be numerically stable. In the paper references above, Hull and Newberry gave as a condition for reducing growth of propagated error, maximizing $\sum B_i$ while minimizing truncation constant E_{n+1}. In our process, $\sum_{i=0}^{4} B_i$ becomes an increasing function of K and $1.89 < \sum_{i=0}^{4} B_i < 3.2$ while $|E_{n+1}| < .0159$. This compares favorably with the fifth order Adams' corrector which has $|E_{n+1}| \approx .0187$ and $\sum_{i=0}^{4} B_i = 1$. If we considered the propagation coefficients to be fixed and equal throughout the process (as is the classical approach), the idea that the propagated error is dominated by the principal root of the propagation equation is equivalent to assuming that the error grows geometrically, i.e., $\varepsilon_{n+1} = (1+r)^{n+1}\varepsilon_0$, where ε_0 is the initial propagated error and $|r| < 1$. Under the same conditions of equal and fixed propagation coefficients (call them α) with the additional restrictions of $\varepsilon_n \approx \varepsilon_{n-1} \approx \varepsilon_{n-2} \approx \varepsilon_E$, $\varepsilon_3 \approx \varepsilon_2 \approx \varepsilon_1 \approx \varepsilon_0 \approx \varepsilon_I$ and truncation error $T = \frac{R}{1-KB_4}$ fixed at each step, the idea of dominance of principal root $(r_1 \approx e^K)$ yields

$$\varepsilon_{n+1} \approx \varepsilon_E[3\alpha + (\alpha-1)e^K + (\alpha-1)e^{2K} + (\alpha-1)e^{3K}]$$
$$+ \varepsilon_I e^{(n-7)K}[4e^{3K} + 3e^{2K} + 2e^K + 1] + T\sum_{i=0}^{n-4} e^{iK} .$$

We now obtain a general expression for the $n+1^{st}$ propagated error of our system by applying methods from Calculus of Finite Differences by Charles Jordan [6, p. 587]

Theorem 3. If we require propagation coefficients to be equal at each step, but allow change from step to step as previously described, then

$$\varepsilon_{n+1} = \sum_{m=0}^{(n+1)-4} \psi(K_{n+1},m)\, T_{(n+1)-m} + \sum_{m=(n+1)-3}^{n+1} \psi(K_{n+1},m)\, \varepsilon_{(n+1)-m} , \tag{15}$$

with

$$\psi(K_{n+1},m) = \sum a_{j_1}(K_{n+1})a_{j_2}(K_{n+1-j_1})a_{j_3}(K_{n+1-j_1-j_2})\cdots$$
$$a_{j_i}(K_{n+1-m+j_i}) ,$$

where

1. $\alpha(K_i) = a_1(K_i) = a_2(K_i) = a_3(K_i) = a_4(K_i)$,
2. $j_1 + j_2 + \ldots + j_i = m$,
3. $j_1, j_2, \ldots, j_i$ are combinations of the numbers, 1,2,3,4,
4. summations are taken over all partitions of m by 1,2,3, and 4 with repetitions and permutations included,

and

$$T_{(n+1)-m} = \frac{K_{(n+1)} - m}{1 - K_{(n+1)-m}\, B\, (K_{n+1)-m})} .$$

Proof: We consider the following system of linear equations with variable coefficients

$$(16) \quad \varepsilon_{n+1} - a_1(K_{n+1})\varepsilon_n - a_2(K_{n+1})\varepsilon_{n-1} - a_3(K_{n+1})\varepsilon_{n-2} - a_4(K_{n+1})\varepsilon_{n-3} = T_{n+1},$$

$$\varepsilon_n - a_1(K_n)\varepsilon_{n-1} - a_2(K_n)\varepsilon_{n-2} - a_3(K_n)\varepsilon_{n-3} - a_4(K_n)\varepsilon_{n-4} = T_n,$$

$$\vdots$$

$$\varepsilon_5 - a_1(K_5)\varepsilon_4 - a_2(K_5)\varepsilon_3 - a_3(K_5)\varepsilon_2 - a_4(K_5)\varepsilon_1 = T_5,$$

$$\varepsilon_4 - a_1(k_4)\varepsilon_3 - a_2(K_4)\varepsilon_2 - a_3(K_4)\varepsilon_1 - a_4(K_4)\varepsilon_0 = T_4,$$

$$\varepsilon_3 - a_1(K_3)\varepsilon_2 - a_2(K_3)\varepsilon_1 - a_3(K_3)\varepsilon_0 = \varepsilon_3,$$

$$\varepsilon_2 - a_1(K_2)\varepsilon_1 - a_2(K_2)\varepsilon_0 = \varepsilon_2,$$

$$\varepsilon_1 - a_1(K_1)\varepsilon_0 = \varepsilon_1,$$

$$\varepsilon_0 = \varepsilon_0,$$

where $a_i(K_t) = \alpha(K_t)$ for all t and $\alpha(K_3) = \alpha(K_2) = \alpha(K_1) = 0$.
Solving the above system of n+2 equation in n+2 unknowns for ε_{n+1} by Cramer's Rule and applying techniques from the above reference, we obtain the desired results.

It should be noted that in the language of combinatorial analysis, the number of terms in the coefficient of $T_{(n+1)-m}$ is precisely the number of compositions of m with no restriction on the number of parts but with no part greater than 4. This is denoted c(m,4).

Corollary 3.1. If the initial propagated errors

$\varepsilon_3 = \varepsilon_2 = \varepsilon_1 = \varepsilon_0 = \varepsilon$, $T_{(n+1)-m} = T$ at each step and

$\overset{*}{\alpha}_m = \max\{\alpha(K_{n+1}), \alpha(K_n), \ldots, \alpha(K_{n+1-(m-1)})\}$,

then

$$|\varepsilon_{n+1}| \leq |T| \sum_{m=0}^{(n+1)-4} c(m,4)\overset{*}{\alpha}_m^{j_m} + |\varepsilon| \sum_{m=(n+1)-3}^{n+1} c(m,4)\overset{*}{\alpha}_m^{j_m},$$

where j_m is the smallest integer satisfying $j_m \geq \frac{m}{4}$.

We obtain another general bound for ε_{n+1} by applying a theorem of Hadamard [1] which states that $|\det A| \leq \pi r_i$ where $r_i = \sqrt{\sum_{k=1}^{n} |a_{ik}|^2}$ and and $A = |a_{ik}|$ is an nXn matrix.

Corollary 3.2. If the initial propagation errors $\varepsilon_0 = \varepsilon_1 = \varepsilon_2 = \varepsilon_3 = \varepsilon$ and $T_{(n+1)-m} = T$ at each step, then

$$|\varepsilon_{n+1}| \leq |T| \left[1 + 3\overset{*}{\alpha} + 2\overset{*}{\alpha}^2 + 2\overset{*}{\alpha}^3 + \sum_{m=4}^{(n+1)-4} 2\overset{*}{\alpha} \cdot (1+2\overset{*}{\alpha})^{m-4}(1+\sqrt{3}\overset{*}{\alpha}) \cdot (1+\sqrt{2}\overset{*}{\alpha}) \cdot (1+\overset{*}{\alpha})\right] + |\varepsilon| \sum_{m=(n+1)-3}^{n+1} 2\overset{*}{\alpha} \cdot (1+2\overset{*}{\alpha})^{m-4} \cdot (1+\sqrt{3}\overset{*}{\alpha})(1+\sqrt{2}\overset{*}{\alpha})(1+\overset{*}{\alpha})$$

where $\overset{*}{\alpha} = \max\{\alpha(K_{n+1}), \alpha(K_n), \ldots, \alpha(K_4)\}$.

Many other bounds may be found with additional restrictions on K and the differential equation being considered.

5. Numerical Results. In order to test our process in actual numerical calculations, we have chosen four differential equations with initial conditions. These choices are supported in the literature [3, p.209] and are designed to test our process for stability and for solving higher order equations and systems of equations. In order to give some idea of the value of our process, we will compare its results with results obtained by using the Adams' corrector alone. The Adams' formulas are widely

recognized as standards by which the stability properties of other processes can be measured. We compare the processes by comparing the two quantities error and relative error. Letting z(m) denote the m^{th} exact value of y, we define

$$\text{Error} = |y(m) - z(m)|$$

and

$$\text{Relative Error} = \left|\frac{\text{Error}}{z(m)}\right|.$$

Many references consider relative error to be more meaningful than error since it takes into consideration the size of the value being calculated. Tables 1 through 8 give values for error and relative error for each equation for stepsizes h = 0.05 and h = 0.01. The next four tables give a comparison of stepwise change of relative error and the final two tables give comparisons of the growth of relative error over the entire range of the calculations.

Although our results were obtained to 15 significant figures, we have rounded the figures in our tables to 2 or 4 places whichever is more meaningful in each case. We denote our process with equal propagation coefficients by EPC. To make the process more realistically useful, we have designed our computer program to approximate $h\frac{\partial f}{\partial y}$ at each step of the calculations instead of feeding in the exact values for this quantity ($.5142 \times 10^{-5}$ is denoted $.5142^{-05}$).

The author wishes to express his sincere appreciation to Professor James R. Wesson for his valuable guidance during this study. Much of this work was supported by NASA contract NAS 8-2559.

BIBLIOGRAPHY

1. Bodewig, E., Matrix Calculus, North-Holland Publishing Co. Amsterdam, 1956,p. 69.MR 18 #235.

2. Dahlquist, Germund, "Stability and Error Bounds in the Numerical Integration of Ordinary Differential Equations", Dissertation, Uppsala, 1958.

3. Hamming, R.W., Numerical Methods for Scientists and Engineers, McGraw-Hill, New York, 1962.MR 25 #735.

4. Hull, T.E. and Newberry, A.C.R., "Corrector Formulas for Multi-Step Integration Methods", J.Soc.Indust. Appl. Math., v.10,1962, pp. 351-369. MR 27 #2130.

5. Hull, T.E. and Newberry, A.C.R., "Integration Procedures Which Minimize Propagated Errors",J.Soc.Indust.Appl. Math., v. 9, 1961,pp. 31-47. MR 22 #11519.

6. Jordan, Charles, Calculus of Finite Differences,Chelsea Publishing Co., New York, 1947.

7. Rall, L.B.(edited), Error In Digital Computation, Wiley, New York, 1965. MR 30 #5510.

8. Ralston,A., "Relative Stability In the Numerical Solution of Ordinary Differential Equations", Soc. Indust. Appl. Math. Rev., v. 7, 1965. pp 114-125. MR 31 #2831.

TABLE 1

COMPARISON OF EPC AND ADAMS' FORMULAS

FOR y' = xy AT h = 0.05.

	EPC		ADAMS'	
x	ERROR	RELATIVE ERROR	ERROR	RELATIVE ERROR
1.00	$.7484^{-07}$	$.4539^{-07}$	$.2547^{-06}$	$.1545^{-06}$
2.00	$.2577^{-05}$	$.3488^{-06}$	$.9241^{-05}$	$.1251^{-05}$
3.00	$.1657^{-03}$	$.1841^{-05}$	$.6150^{-03}$	$.6832^{-05}$
4.00	$.2204^{-01}$	$.7394^{-05}$	$.8427^{-01}$	$.2827^{-04}$
5.00	$.6451^{+01}$	$.2404^{-04}$	$.2532^{+02}$	$.9435^{-04}$
6.00	$.4347^{+04}$	$.6620^{-04}$	$.1747^{+05}$	$.2661^{-03}$
7.00	$.6976^{+07}$	$.1597^{-03}$	$.2867^{+08}$	$.6565^{-03}$
8.00	$.2735^{+11}$	$.3464^{-03}$	$.1149^{+12}$	$.1455^{-02}$
9.00	$.2672^{+15}$	$.6884^{-03}$	$.1146^{+16}$	$.2953^{-02}$
10.00	$.6594^{+19}$	$.1272^{-02}$	$.2891^{+20}$	$.5577^{-02}$

TABLE 2

COMPARISON OF EPC AND ADAMS' FORMULAS

FOR $y' = xy$ AT $h = 0.01$

	EPC		ADAMS'	
x	ERROR	RELATIVE ERROR	ERROR	RELATIVE ERROR
1.00	$.2606^{-10}$	$.1581^{-10}$	$.9797^{-10}$	$.5942^{-10}$
2.00	$.9240^{-09}$	$.1250^{-09}$	$.3507^{-08}$	$.4746^{-09}$
3.00	$.6266^{-07}$	$.6961^{-09}$	$.2395^{-06}$	$.2661^{-08}$
4.00	$.8824^{-05}$	$.2960^{-08}$	$.3394^{-04}$	$.1138^{-07}$
5.00	$.2737^{-02}$	$.1020^{-07}$	$.1059^{-01}$	$.3945^{-07}$
6.00	$.1957^{+01}$	$.2981^{-07}$	$.7607^{+01}$	$.1158^{-06}$
7.00	$.3338^{+04}$	$.7642^{-07}$	$.1303^{+05}$	$.2984^{-06}$
8.00	$.1393^{+08}$	$.1764^{-06}$	$.5462^{+08}$	$.6917^{-06}$
9.00	$.1450^{+12}$	$.3735^{-06}$	$.5709^{+12}$	$.1471^{-05}$
10.00	$.3820^{+16}$	$.7368^{-06}$	$.1510^{+17}$	$.2913^{-05}$

TABLE 3

COMPARISON OF EPC AND ADAMS' FORMULAS

FOR $y' = -xy$ AT $h = 0.05$

	EPC		ADAMS'	
x	ERROR	RELATIVE ERROR	ERROR	RELATIVE ERROR
1.00	$.1676^{-08}$	$.2764^{-08}$	$.8655^{-08}$	$.1427^{-07}$
2.00	$.3195^{-08}$	$.2361^{-07}$	$.1237^{-07}$	$.9139^{-07}$
3.00	$.1185^{-08}$	$.1067^{-06}$	$.3838^{-08}$	$.3455^{-06}$
4.00	$.1177^{-09}$	$.3507^{-06}$	$.2985^{-09}$	$.8898^{-06}$
5.00	$.2974^{-10}$	$.7980^{-05}$	$.9386^{-10}$	$.2519^{-04}$
6.00	$.7273^{-12}$	$.4776^{-04}$	$.2363^{-11}$	$.1552^{-03}$
7.00	$.4344^{-14}$	$.1897^{-03}$	$.1437^{-13}$	$.6278^{-03}$
8.00	$.7613^{-17}$	$.6011^{-03}$	$.2575^{-16}$	$.2033^{-02}$
9.00	$.4241^{-20}$	$.1646^{-02}$	$.1485^{-19}$	$.5763^{-02}$
10.00	$.7858^{-24}$	$.4074^{-02}$	$.2921^{-23}$	$.1514^{-01}$

TABLE 4

COMPARISON OF EPC AND ADAMS' FORMULAS

FOR $y' = -xy$ AT $h = 0.01$

	EPC		ADAMS'	
x	ERROR	RELATIVE ERROR	ERROR	RELATIVE ERROR
1.00	$.7255^{-12}$	$.1279^{-11}$	$.3153^{-11}$	$.5199^{-11}$
2.00	$.8487^{-10}$	$.6271^{-11}$	$.3290^{-11}$	$.2431^{-10}$
3.00	$.3177^{-12}$	$.2860^{-10}$	$.1188^{-11}$	$.1070^{-09}$
4.00	$.2937^{-13}$	$.8754^{-10}$	$.1042^{-12}$	$.3106^{-09}$
5.00	$.6993^{-14}$	$.1876^{-08}$	$.2587^{-13}$	$.6943^{-08}$
6.00	$.1000^{-15}$	$.1051^{-07}$	$.5946^{-15}$	$.3904^{-07}$
7.00	$.8899^{-18}$	$.3886^{-07}$	$.3307^{-17}$	$.1444^{-06}$
8.00	$.1442^{-20}$	$.1139^{-06}$	$.5357^{-20}$	$.4230^{-06}$
9.00	$.7368^{-24}$	$.2860^{-06}$	$.2732^{-23}$	$.1060^{-05}$
10.00	$.1238^{-27}$	$.6416^{-06}$	$.4580^{-27}$	$.2374^{-05}$

TABLE 5

COMPARISON OF EPC AND ADAMS' FORMULAS

FOR $y' = -2xy^2$ AT $h = 0.05$

	EPC		ADAMS'	
x	ERROR	RELATIVE ERROR	ERROR	RELATIVE ERROR
1.00	$.4972^{-07}$	$.9945^{-07}$	$.2118^{-06}$	$.4237^{-06}$
2.00	$.3840^{-09}$	$.1920^{-08}$	$.3508^{-08}$	$.1754^{-07}$
3.00	$.3221^{-09}$	$.3221^{-08}$	$.1613^{-08}$	$.1613^{-07}$
4.00	$.1913^{-09}$	$.3252^{-08}$	$.8701^{-09}$	$.1479^{-07}$
5.00	$.9983^{-10}$	$.2596^{-08}$	$.4432^{-09}$	$.1152^{-07}$
6.00	$.5399^{-10}$	$.1998^{-08}$	$.2374^{-09}$	$.8784^{-08}$
7.00	$.3101^{-10}$	$.1550^{-08}$	$.1357^{-09}$	$.6784^{-08}$
8.00	$.1885^{-10}$	$.1225^{-08}$	$.8227^{-10}$	$.5348^{-08}$
9.00	$.1204^{-10}$	$.9877^{-09}$	$.5248^{-10}$	$.4303^{-08}$
10.00	$.8025^{-11}$	$.8106^{-09}$	$.3493^{-10}$	$.3528^{-08}$

TABLE 6

COMPARISON OF EPC AND ADAMS' FORMULAS

FOR $y' = -2xy^2$ AT h = 0.01

	EPC		ADAMS'	
x	ERROR	RELATIVE ERROR	ERROR	RELATIVE ERROR
1.00	$.8937^{-11}$	$.1787^{-10}$	$.3631^{-10}$	$.7261^{-10}$
2.00	$.6837^{-12}$	$.3419^{-11}$	$.2446^{-11}$	$.1223^{-10}$
3.00	$.1040^{-12}$	$.1040^{-11}$	$.3685^{-12}$	$.3685^{-11}$
4.00	$.1165^{-13}$	$.1980^{-12}$	$.3598^{-13}$	$.6117^{-12}$
5.00	$.8535^{-15}$	$.2219^{-13}$	$.5120^{-14}$	$.1331^{-12}$
6.00	$.2015^{-14}$	$.7593^{-13}$	$.7780^{-14}$	$.2731^{-12}$
7.00	$.1684^{-14}$	$.8418^{-13}$	$.5060^{-14}$	$.2530^{-12}$
8.00	$.1319^{-14}$	$.8575^{-13}$	$.2820^{-14}$	$.1833^{-12}$
9.00	$.9151^{-15}$	$.7504^{-13}$	$.1210^{-14}$	$.9922^{-13}$
10.00	$.6661^{-15}$	$.6728^{-13}$	$.1700^{-15}$	$.1717^{-13}$

TABLE 7

COMPARISON OF EPC AND ADAMS' FORMULAS

FOR $y'' = -(xy'+y)/(xy)^2$ AT h=0.05

	EPC		ADAMS'	
x	ERROR	RELATIVE ERROR	ERROR	RELATIVE ERROR
4.00	$.7836^{-05}$	$.4033^{-05}$	$.2135^{-04}$	$.1099^{-04}$
5.00	$.5153^{-04}$	$.1956^{-04}$	$.1432^{-03}$	$.5456^{-04}$

TABLE 8

COMPARISON OF EPC AND ADAMS' FORMULAS

FOR $y'' = -(xy'+y)/(xy)^2$ AT h = 0.01

	EPC		ADAMS'	
x	ERROR	RELATIVE ERROR	ERROR	RELATIVE ERROR
4.00	$.3215^{-07}$	$.1655^{-07}$	$.1600^{-07}$	$.8237^{-08}$
19.00	$.1974^{-06}$	$.7518^{-07}$	$.1034^{-06}$	$.3941^{-07}$

TABLE 9

COMPARISON OF STEPWISE GROWTH OF RELATIVE

ERROR OF EPC AND ADAMS' FORMULAS FOR $y' = xy$.

	h = 0.05		h = 0.01	
x	EPC	ADAMS'	EPC	ADAMS'
1.00-2.00	7.68	8.10	7.91	7.99
2.00-3.00	5.28	5.46	5.57	5.61
3.00-4.00	4.02	4.14	4.25	4.28
4.00-5.00	3.25	3.34	3.45	3.47
5.00-6.00	2.75	2.82	2.92	2.94
6.00-7.00	2.41	2.47	2.56	2.58
7.00-8.00	2.17	2.22	2.31	2.32
8.00-9.00	1.99	2.03	2.10	2.13
9.00-10.00	1.85	1.89	1.97	1.98

TABLE 10

COMPARISON OF STEPWISE GROWTH OF RELATIVE

ERROR OF EPC AND ADAMS' FORMULAS FOR $y' = -xy$

	h = 0.05		h = 0.01	
x	EPC	ADAMS'	EPC	ADAMS'
1.00-2.00	8.54	6.40	4.90	4.68
2.00-3.00	4.52	3.78	4.56	4.40
3.00-4.00	3.29	2.58	3.06	2.90
4.00-5.00	22.75	28.31	21.43	22.35
5.00-6.00	5.99	6.16	5.60	5.62
6.00-7.00	3.97	4.05	3.70	3.70
7.00-8.00	3.17	3.24	2.93	2.93
8.00-9.00	2.74	2.83	2.51	2.51
9.00-10.00	2.48	2.63	2.24	2.24

TABLE 11

COMPARISON OF STEPWISE GROWTH OF RELATIVE ERROR OF EPC AND ADAMS' FORMULAS FOR $y' = -2xy^2$

	h = 0.05		h = 0.01	
x	EPC	ADAMS'	EPC	ADAMS'
1.00-2.00	.0193	.0414	.1913	.1684
2.00-3.00	1.6776	.9196	.3040	.3013
3.00-4.00	1.0096	.9169	.1904	.1660
4.00-5.00	.7983	.7789	.1121	.2176
5.00-6.00	.7696	.7625	3.4218	2.0518
6.00-7.00	.7758	.7703	1.1097	.9264
7.00-8.00	.7903	.7883	1.0187	.7245
8.00-9.00	.8063	.8046	.8751	.5413
9.00-10.00	.8207	.8199	.8966	.1731

TABLE 12

GROWTH OF RELATIVE ERROR FROM x = 1.00 TO x = 10.00 FOR EPC AND ADAMS' FORMULAS

	h = 0.05		h = 0.01	
	EPC	ADAMS'	EPC	ADAMS'
xy	$.2802^{+05}$	$.3610^{+05}$	$.4660^{+05}$	$.4900^{+05}$
$-xy$	$.1474^{+07}$	$.1061^{+07}$	$.5016^{+06}$	$.4566^{+06}$
$-2xy^2$	$.8151^{-02}$	$.8327^{-02}$	$.3765^{-02}$	$.2365^{-03}$

TABLE 13

GROWTH OF RELATIVE ERROR OF EPC AND ADAMS'

FORMULAS FROM x = 4.00 TO x = 19.00 FOR

$$y'' = -(xy'+y)/(xy)^2$$

h = 0.05		h = 0.01	
EPC	ADAMS'	EPC	ADAMS'
$.4850^{+01}$	$.4965^{+01}$	$.4543^{+01}$	$.4785^{+01}$

ITERATIVE SOLUTION OF ABSTRACT POLYNOMIAL EQUATIONS

L. B. Rall*
Mathematics Research Center
University of Wisconsin
Madison, Wisconsin

ABSTRACT. The nonlinearities in many equations of practical importance are polynomial in character, viewed from the abstract standpoint of multilinear operators in Banach spaces. After appropriate definitions and theorems on the properties of such operators, a standard class of abstract polynomial equations is identified. For equations in this class, existence and uniqueness of solutions may be established by simple calculations with appropriate scalar majorant polynomials. These results also give conditions for the convergence of the method of successive substitution (simple iteration), Newton's method, and the modified form of Newton's method to a solution, including numerical values for error estimates. For the class of abstract polynomial equations considered, the method of successive substitutions and the modified form of Newton's method are shown to be identical.

The theory is illustrated by application to finite polynomial systems, and to particular polynomial integral and differential equations.

1. INTRODUCTION. The nonlinear operator equations considered in this paper are natural generalizations of scalar polynomial equations to the more abstract setting of Banach spaces. This class of abstract polynomial equations includes a number of interesting differential and integral equations, which contain nonlinearities consisting of powers or products of the unknown functions, mingled perhaps with linear differential or integral operators. A theory of abstract polynomial equations of completeness comparable to the theory of scalar polynomial equations would be highly desirable, but is not available at the present time. After framing suitable definitions, the present work will proceed to the construction of local theories for a specific class of abstract polynomial equations, the ones which will be called regular at some point x_0 of the Banach space X. These local theories will give information concerning the existence, uniqueness, and construction of a solution in a neighborhood of x_0. The methods used are based on the well-known theorems for the solution of operator equations by the technique of successive substitutions (simple iteration) and by Newton's method [13]. The requisite facts for the abstract polynomial equation are deduced from the corresponding result for a scalar majorant polynomial, which can be obtained by fairly simple direct calculations.

*Mathematics Research Center, University of Wisconsin, Madison 53706. Sponsored by the United States Army under Contract No. DA-31-124-ARO-D-462. The remainder of this paper has been reproduced photographically from the author's manuscript.

2. Definitions and notation. Let X and Y denote linear spaces (not necessarily distinct) over a common scalar field Λ, which will be restricted to be the real or complex number system. The space of linear (additive and homogeneous) operators L from X into Y will be denoted by $\mathfrak{L}(X,Y)$. A linear operator B from X into $\mathfrak{L}(X,Y)$ is called a bilinear operator from X into Y, for

$$Bx_1x_2 = (Bx_1)x_2 \tag{2.1}$$

is an element of Y for all $x_1, x_2 \in X$. It follows that the bilinear form (2.1) is linear in each argument. For $x_1 = x_2 = x$, the nonlinear mapping from X into Y defined by

$$y = Bxx \tag{2.2}$$

is a simple generalization of multiplying the square of a scalar variable by a constant coefficient [11]. By a simple induction [13, pp. 100-108], one may define multilinear operators and polynomial operators of arbitrary degree. For $n = 2, 3, \ldots$, let $\mathfrak{L}(X^n, Y)$ denote the set of linear operators N from X into $\mathfrak{L}(X^{n-1}, Y)$. The operator N will be called an n-linear operator from X into Y.

If $N \in \mathfrak{L}(X^n, Y)$, and n points $x_1, x_2, \ldots, x_n \in X$ are given, then

$$y = Nx_1x_2\ldots x_n \tag{2.3}$$

will be a point of Y, the convention being that N operates on x_1, the $(n-1)$-linear operator Nx_1 operates on x_2, and so on. In general, the order of operation is important. For a permutation $\iota = (i_1, i_2, \ldots, i_n)$ of the integers $1, 2, \ldots, n$, the notation $N(\iota)$ will be used for the n-linear operator from X into Y such that

(2.4) $$N(\iota)x_1x_2\ldots x_n = Nx_{i_1}x_{i_2}\ldots x_{i_n}$$

for all $x_1, x_2, \ldots, x_n \in X$.

Thus, there are $n!$ n-linear operators $N(\iota)$ associated with a given n-linear operator N.

An n-linear operator N from X into Y is said to be <u>symmetric</u> if

(2.5) $$N = N(\iota)$$

for all $\iota \in \Pi_n$, where Π_n denotes the set of all permutations of the integers $1, 2, \ldots, n$. The symmetric n-linear operator

(2.6) $$\overline{N} = \frac{1}{n!} \sum_{\iota \in \Pi_n} N(\iota)$$

is called the <u>mean</u> of N.

An alternative definition of symmetry of an n-linear operator N would be to require that $N = \overline{N}$. A simple nonlinear operator from X into Y is obtained by taking $x_1 = x_2 = \ldots = x_n = x$ in (2.4). The result is an obvious generalization of the product of the nth power of a scalar variable and a constant coefficient.

It will be convenient at times to use the notation

(2.7) $$Nx^m = N\overbrace{x\ldots x}^{m},$$

$m \leq n$, $N \in \mathcal{L}(X^n, Y)$, for the result of applying N to $x \in X$ m times.

If $m < n$, then (2.7) will represent an $(n-m)$-linear operator from X into Y. For the special case $m = n$, note that

(2.8) $$Nx^n = \overline{N}x^n = N(\iota)x^n$$

for all $\iota \in \Pi_n$, $x \in X$.

For $A_i \in \mathcal{L}(X^i, Y)$, $i = 1, 2, \ldots, n$ and $A_0 \in Y$, the operator P defined by

$$P(x) = A_n x^n + \ldots + A_2 x^2 + A_1 x + A_0 \tag{2.9}$$

is called an abstract polynomial operator of degree n from X into Y. The equation

$$P(x) = 0 \tag{2.10}$$

is called an abstract polynomial equation of degree n.

It follows from (2.8) that the multilinear operators $A_2, A_3, \ldots, A_n$ in (2.19) may be assumed to be symmetric without loss of generality, since each A_i in (2.9) may be replaced by $\overline{A}_i$, $i = 2, 3, \ldots, n$, without changing the value of $P(x)$. Unless the contrary is explicitly stated, the multilinear operators in all abstract polynomials considered henceforth will be assumed to be symmetric.

The operator

$$P^{(m)}(x) = n(n-1)\ldots(n-m+1)A_n x^{n-m} + (n-1)(n-2)\ldots(n-m-2)A_{n-1}x^{n-m-1} + \ldots + m!A_m \tag{2.11}$$

is called the mth derivative of the abstract polynomial operator P, $m = 1, 2, \ldots, n$.

Note that $P^{(m)}(x) \in \mathcal{L}(X^m, Y)$ for $m = 1, 2, \ldots, n$, and that $P''(x)$, $P'''(x), \ldots, P^{(n)}(x)$ are symmetric multilinear operators. The computation of $P(x)$ and its derivatives at a point $x = x_0$ may be accomplished by adapting Horner's algorithm for scalar polynomials to this purpose [13, p. 111]. An algebraic formulation of this algorithm may be obtained by setting

$$(2.12)\qquad \begin{cases} A_i^{(0)} = A_i\,, & i = 0,1,\dots,n\,, \\ A_n^{(j)} = A_n\,, & j = 1,2,\dots,n+1\,, \end{cases}$$

and calculating

$$(2.13)\qquad \begin{cases} A_{n-k}^{(j+1)} = A_{n-k+1}^{(j+1)} x_0 + A_{n-k}^{(j)}\,, \\ j = 0,1,\dots,n-1\,;\ k = 1,2,\dots,n-j\,. \end{cases}$$

The results of this calculation are

$$(2.14)\qquad A_j^{(j+1)} = \frac{1}{j!} P^{(j)}(x_0)\,,$$

$j = 0,1,\dots,n$, the notation $P^{(0)}(x_0)$ being used for $P(x_0)$.

Remark 2.1. Taylor's identity

$$(2.15)\qquad P(x) = P(x_0) + P'(x_0)(x-x_0) + \tfrac{1}{2}P''(x_0)(x-x_0)^2 + \dots + \frac{1}{n!}P^{(n)}(x_0)(x-x_0)^n$$

holds at any $x_0 \in X$ [13, p. 111].

An abstract polynomial operator P is said to be regular at x_0 if the linear operator $P'(x_0)$ is one-to-one and onto Y, so that the (left) inverse $[P'(x_0)]^{-1}$ exists.

If P is regular at x_0, then one may set

$$(2.16)\qquad \begin{cases} B_i = (i!)^{-1}[P'(x_0)]^{-1} P^{(i)}(x_0)\,, & i = 1,2,\dots,n\,, \\ h = x - x_0\,, \\ B_0 = [P'(x_0)]^{-1} P(x_0)\,, \end{cases}$$

to obtain the abstract polynomial in X,

$$(2.17)\qquad R(h) = B_n h^n + \dots + B_2 h^2 + h + B_0\,.$$

The problem of solving the polynomial equation $P(x) = 0$ is equivalent to solution of the polynomial equation

$$R(h) = 0 ,$$

provided that $P(x)$ is regular at x_0.

Henceforth, attention will be restricted to polynomial equations of the form (1.1).

Remarks for polynomial operators in Banach spaces. Henceforth, X and Y will be assumed to be Banach spaces over Λ, that is, complete normed linear spaces. Since confusion is unlikely, the norm in either space will be denoted by $\|\cdot\|$. Considering only bounded operators, the spaces $\mathcal{L}(X^n,Y)$, $n = 1, 2, \ldots$, will also be Banach spaces [7] for the norm

$$\|N\| = \sup_{\|x\|=1} \|Nx\| .$$

In the case $n = 1$, N will simply be a linear operator from X into Y.

If $N \in \mathcal{L}(X^n,Y)$ and $m \leq n$, then

$$\|Nx^m\| \leq \|N\| \cdot \|x\|^m .$$

An abstract polynomial operator P from X into Y of degree n defined by

$$P(x) = A_n x^n + \ldots + A_2 x^2 + A_1 x + A_0 ,$$

is said to be bounded if its coefficients A_i, $i = 1, 2, \ldots, n$, are bounded multilinear operators from X into Y.

If

$$a_i \geq \|A_i\| ,$$

$i = 1, \ldots, n$, the real polynomial

$$p(r) = a_n r^n + \ldots + a_2 r^2 + a_1 r + a_0$$

is called a scalar majorant polynomial for the bounded abstract polynomial operator (3.3).

If (3.5) is a scalar majorant polynomial for the operator defined by (3.3), then for $\|x\| \leq r$,

$$\|P(x)\| \leq p(r), \tag{3.6}$$

$$\|P^{(i)}(x)\| \leq p^{(i)}(r), \qquad i = 1, 2, \ldots, n, \tag{3.7}$$

and

$$\|P^{(n+j)}(x)\| = p^{(n+j)}(r) = 0, \qquad j = 1, 2, \ldots. \tag{3.8}$$

The formal differentiation process defined in §2 when applied to bounded abstract polynomial operators yields their Fréchet derivatives [13, pp. 108-115]. In general, the mth Fréchet derivative of an operator F from X into Y will be bounded, symmetric m-linear operator from X into Y [7, 11]. For operators which have a continuous mth Fréchet derivative, the following estimate holds:

$$\|F(x) - \sum_{k=0}^{m-1} \frac{1}{k!} F^{(k)}(x_0)(x - x_0)^k\| \leq \tag{3.9}$$

$$\leq \frac{1}{m!} \sup_{\bar{x} \in [x_0, x]} \|F^{(m)}(\bar{x})\| \, \|x - x_0\|^m,$$

where

$$[x_0, x] = \{\bar{x} : \bar{x} = \theta x + (1 - \theta)x_0, \ 0 \leq \theta \leq 1\} \tag{3.10}$$

is the line segment from x_0 to x [13, p. 125; 6, 8].

Theorem 3.1. If $P(x)$ is a bounded abstract polynomial operator, and $p(r)$ a scalar majorant polynomial for it, then

$$(3.11)\qquad \|P(x) - \sum_{k=0}^{m-1} \frac{1}{k!} P^{(k)}(x_0)(x-x_0)^k\| \leq \frac{p^{(m)}(R)}{m!} \|x-x_0\|^m ,$$

where

$$(3.12)\qquad R = \max\{\|x\|, \|x_0\|\} .$$

Proof: This follows directly from (3.6)-(3.9), and the fact that

$$(3.13)\qquad \|\bar{x}\| = \|\theta x + (1-\theta)x_0\| \leq \theta R + (1-\theta)R = R .$$

In the following sections, results obtained by considering scalar majorant polynomials will be used to survey various iterative methods for solving abstract polynomial equations.

4. Solution of abstract polynomial equations by successive substitutions. It will be assumed that the bounded abstract polynomial operator P is regular at some $x_0 \in X$, and the transformations (2.16) have been performed to put the polynomial equation

$$(4.1)\qquad P(x) = 0$$

into the form

$$(4.2)\qquad B_n h^n + \ldots + B_2 h^2 + h + B_0 = 0$$

in the space X. Ordinarily, one would be motivated to choose x_0 as an initial approximation to a solution x^* of (4.1), and then seek a solution $h = h^*$ of (4.2) in the vicinity of the origin 0 of X. Of course, the finding of a suitable x_0 may be a significant problem in its own right. A general prescription for finding x_0 would be equivalent to a complete solution of the program of solving abstract polynomial equations, and is not available at the present time. However, one usually has some useful information about the specific problem under consideration, or can find x_0 by some approximation procedure. For example, by choosing a

basis $x_1, x_2, \ldots, x_m$ for a finite-dimensional subspace X_m of X (which may itself be of finite dimension $> m$), one may try to find constants $\alpha_1, \alpha_2, \ldots, \alpha_m$ such that $x_0 = \alpha_1 x_1 + \alpha_2 x_2 + \ldots + \alpha_m x_m$ minimizes $\|P(x)\|$, or, perhaps better, $\|[P(x)]^{-1} P(x)\|$. If m is sufficiently small, this problem may be computationally tractable. In the following, it will be assumed that $n \geq 2$, so that (4.2) will be nonlinear.

One commonly used technique for the solution of equations is the method of <u>successive substitutions</u> (or <u>iteration</u>). Equation (4.2) is written in the form

$$h = F(h), \tag{4.3}$$

where

$$F(h) = -(B_n h^n + \ldots + B_2 h^2 + B_0). \tag{4.4}$$

Solutions of (4.3) are called <u>fixed points</u> of the operator F. The form of (4.3) suggests the iteration

$$h_0 = 0, \quad h_{k+1} = F(h_k), \tag{4.5}$$

$k = 0, 1, 2, \ldots$. If the sequence $\{h_k\}$ converges to $h = h^*$, then h^* will be a solution of (4.3), and hence of (4.2). The point

$$x^* = x_0 + h^* \tag{4.6}$$

will then satisfy the original polynomial equation (4.1).

Conditions for the convergence of this process are given by a famous theorem of Banach [1] and Caccioppoli [2].

Theorem 4.1. If a non-negative constant $\mu < 1$ exists such that

$$\|F(x) - F(y)\| \leq \mu \|x - y\| \tag{4.7}$$

for all $x, y \in \overline{U}(r)$, where

$$\overline{U}(r) = \{x : \|x\| \leq r\}, \tag{4.8}$$

and

$$r \geq \frac{\eta_0}{1-\mu} \,, \tag{4.9}$$

where

$$\eta_0 = \|h_1 - h_0\| = \|B_0\| \,, \tag{4.10}$$

then the sequence $\{h_k\}$ generated by (4.5) converges to a fixed point h^* of Γ which is unique in $\overline{U}(r)$, with

$$\|h^* - h_k\| \leq \frac{\mu^k}{1-\mu}\eta_0 \,, \tag{4.11}$$

$k = 0, 1, 2, \ldots$.

This theorem is simple to prove [13], and gives information concerning the existence of a solution h^* of (4.3) near the origin 0 of X, and the error bound (4.11) for the terms of the sequence $\{h_k\}$ as approximations to h^*.

For the present purposes, it is required to obtain a bound μ for the operator defined by (4.4). If

$$b_i \geq \|B_i\| \,, \qquad i = 2, 3, \ldots, n \,, \tag{4.12}$$

then $\Gamma(h)$ has the scalar majorant polynomial

$$f(r) = b_n r^n + \ldots + b_2 r^2 + \eta_0 \tag{4.13}$$

for $h \in \overline{U}(r)$. Similarly,

$$\|\Gamma'(h)\| \leq f'(r) = n b_n r^{n-1} + \ldots + 2 b_2 r \tag{4.14}$$

if $h \in \overline{U}(r)$. By Theorem 3.1 (see inequality (3.11)),

$$\|\Gamma(x) - \Gamma(y)\| \leq f'(r)\|x - y\| \tag{4.15}$$

for $x, y \in \overline{U}(r)$. From (4.14), $f'(0) = 0$, and for $r > 0$, $f'(r)$ is a positive, strictly monotone increasing function which goes to infinity as $r \to +\infty$.

Let $r = R$ be such that

(4.16) $$f'(R) = 1 .$$

Then, for $0 \leq r < R$, the function

(4.17) $$r_0(r) = \frac{\eta_0}{1 - f'(r)}$$

is positive (assuming $\eta_0 > 0$), strictly convex and monotone increasing, and goes to $+\infty$ as $r \to R$.

Under the conditions given, the curve defined by (4.17) will intersect the line $r_0 = r$ in at most two points r_e, r_u, which may be coincident. These points are determined by the equation

(4.18) $$r = \frac{\eta_0}{1 - f'(r)} ,$$

or, by (4.14),

(4.19) $$nb_n r^n + \ldots + 2b_2 r^2 - r + \eta_0 = 0 .$$

By Descartes' rule of signs [5], (4.19) has two or no positive solutions, which establishes the assertion made above. If r_e, r_u exist, $r_e \leq r_u$, then it is evident that (4.9) is satisfied for r such that

(4.20) $$r_e \leq r \leq r_u .$$

The values of r_e, r_u may be determined as accurately as necessary by solving the simple scalar polynomial equation (4.19). This gives the following result.

Theorem 4.2. If positive solutions $r_e \leq r_u$ of equation (4.19) exist, then a solution $h^* \in \overline{U}(r_e)$ of equation (4.3) exists and is unique in $\overline{U}(r_u)$. Furthermore, the sequence $\{h_k\}$ defined by (4.5) converges to h^*, with

(4.21) $$\|h^* - h_k\| \leq \frac{[f'(r_e)]^k \eta_0}{1 - f'(r_e)} ,$$

$k = 0, 1, 2, \ldots$.

This result gives a simple, computationally verifiable condition for the existence and uniqueness of a solution of the abstract polynomial equation (4.1) in the vicinity of a point x_0 at which $P(x)$ is regular.

For (4.2) quadratic ($n = 2$), equation (4.19) becomes

$$2b_2 - r + \eta_0 = 0 , \tag{4.22}$$

which has real solutions if and only if

$$1 - 8b_2\eta_0 \geq 0 , \tag{4.23}$$

or

$$\|B_0\| \leq \frac{1}{8\|B_2\|} . \tag{4.24}$$

5. Solution of abstract polynomial equations by Newton's method. Another frequently used approach to the solution of nonlinear operator equations is Newton's method in the generality obtained by L. V. Kantorovič [8, 13, 14]. For

$$R(h) = B_n h^n + \ldots + B_2 h^2 + h + B_0 , \tag{5.1}$$

one has the Fréchet derivatives

$$R'(h) = nB_n h^{n-1} + \ldots + 2B_2 h + I , \tag{5.2}$$

where I denotes the identity operator in X, and

$$R''(h) = n(n-1)B_n h^{n-2} + \ldots + 2B_2 . \tag{5.3}$$

Newton's method for finding a solution h^* of

$$R(h) = 0 \tag{5.4}$$

in the vicinity of $h_0 = 0$ consists of formation of the sequence $\{h_k\}$ defined by

$$h_0 = 0 , \quad h_{k+1} = h_k - [R'(h_k)]^{-1} R(h_k) , \tag{5.5}$$

$k = 0, 1, 2, \ldots$, if this is possible.

Since

(5.6) $$R'(h_0) = R'(0) = I ,$$

$[R'(h_0)]^{-1}$ exists, and

(5.7) $$\|[R'(h_0)]^{-1}\| = \|I\| = 1 .$$

From (5.5), one finds that

(5.8) $$h_1 = -R(h_0) = -R(0) = -B_0 ,$$

and

(5.9) $$\eta_0 = \|h_1 - h_0\| = \|B_0\|$$

as before. If now

(5.10) $$\|R''(h)\| \leq K$$

for $h \in \overline{U}(r)$, then the following theorem holds.

Theorem 5.1 (Kantorovič-Ostrowski). If

(5.11) $$K\eta_0 \leq \tfrac{1}{2} ,$$

and

(5.12) $$r \geq r_e = \frac{1 - \sqrt{1 - 2K\eta_0}}{K} ,$$

then a solution h^* of (5.4) exists in $\overline{U}(r_e)$, and the sequence $\{h_k\}$ defined by (5.5) exists and converges to h^*, with

(5.13) $$\begin{cases} \|h^* - h_k\| \leq e^{-2^k\varphi} \dfrac{\sinh \varphi}{\sinh 2^k\varphi} \eta_0 , \\ \dfrac{1}{K\eta_0} = 1 + \cosh \varphi . \end{cases}$$

If (5.11) holds and

(5.14) $$r \geq r_u = \frac{1 + \sqrt{1 - 2K\eta_0}}{K} ,$$

then h^* is unique in $U(r_u) = \{x : \|x\| < r\}$ if $K\eta_0 < \frac{1}{2}$, or is unique in $\overline{U}\left(\frac{1}{K}\right) - \overline{U}(2\eta_0)$ if $K\eta_0 = \frac{1}{2}$.

Proofs of this theorem may be found in the literature [8, 10, 13, 14].

In the present case,

$$f''(r) = n(n-1)b_n r^{n-2} + \ldots + 6b_3 r + 2b_2 \tag{5.15}$$

is a scalar majorant polynomial for $R''(h)$ (see (4.12)-(4.14)).

Here, one looks for intersections of the curve

$$r_0 = \eta_0 f''(r) \tag{5.16}$$

with the curve

$$r_0 = \frac{2\eta_0(r - \eta_0)}{r^2}, \tag{5.17}$$

which will be solutions of the equation

$$r''(r)r^2 - 2r + 2\eta_0 = 0, \tag{5.18}$$

or, from (5.15),

$$n(n-1)b_n r^n + \ldots + 2b_2 r^2 - 2r + 2\eta_0 = 0. \tag{5.19}$$

By Descartes' rule of signs, equation (5.19) has two positive solutions $r_e \leq r_u$ or none. If r_e, r_u exist, then it is evident that $\eta_0 < r_e \leq 2\eta_0$, and that (5.11)-(5.12) are satisfied with

$$K = K_e = f''(r_e). \tag{5.20}$$

Also, if $r_u \geq 2\eta_0$, then (5.14) will be satisfied with

$$K = K_u = f''(r_u). \tag{5.21}$$

This establishes the following result.

Theorem 5.2. If positive solutions $r_e \leq r_u$ of equation (5.19) exist, then equation (5.4) has a solution $h^* \in \overline{U}(r_e)$, to which the sequence $\{h_k\}$ defined by (5.5) converges to h^*, with

$$\begin{cases} \|h^* - h_k\| \leq e^{-2^k \varphi_e} \dfrac{\sinh \varphi_e}{\sinh 2^k \varphi_e} \eta_0, \\ \dfrac{1}{f''(r_e)\eta_0} = 1 + \cosh \varphi_e \end{cases} \tag{5.22}$$

$k = 0, 1, 2, \ldots$. If $r_u > 2\eta_0$, then h^* is unique in $U(r_u)$. If $r_u = 2\eta_0$, then h^* is unique in $\overline{U}(2\eta_0)$.

Since the polynomial on the left side of (5.19) is positive for $r = 0$, and goes to $+\infty$ as $r \to +\infty$, a simple sufficient condition for the existence of r_e, r_u is that it be nonpositive at $r = 2\eta_0$. This gives the following theorem.

Theorem 5.3. If

$$n(n-1)2^{n-1}b_n\eta_0^{n-1} + \ldots + 4b_2\eta_0 \leq 1 , \tag{5.23}$$

then a solution $h = h^*$ of the abstract polynomial equation (5.4) exists and is unique in $\overline{U}(2\eta_0)$, and the sequence $\{h_k\}$ defined by (5.5) converges to h^* .

Applying condition (5.23) to quadratic equations, one obtains

$$4b_2\eta_0 \leq 1 , \tag{5.24}$$

or

$$\|B_0\| \leq \frac{1}{4\|B_2\|} . \tag{5.25}$$

Comparison of (5.25) with (4.24) seems to indicate that Newton' method is far superior to the method of successive substitutions for the solution of abstract quadratic equations. Actually, this is an illusion, as will be shown in the next section.

Condition (5.23) may also be applied to scalar polynomial equations. For example, if

$$R(h) = \frac{h^4}{100} - \frac{h^3}{50} + \frac{h^2}{10} + \frac{1}{2} \tag{5.26}$$

one has

$$b_4 = \frac{1}{100}, \; b_3 = \frac{1}{50}, \; b_2 = \frac{1}{10}, \; \eta_0 = \frac{1}{2} , \tag{5.27}$$

and

(5.28) $$\frac{72}{100}\left(\frac{1}{2}\right)^3 + \frac{24}{50}\left(\frac{1}{2}\right)^2 + \frac{4}{10}\left(\frac{1}{2}\right) = \frac{41}{100} < 1 ,$$

so that (5.23) is satisfied, and the equation $R(h) = 0$ consequently has a unique solution in the interval $-1 \le h \le 1$, to which Newton's method converges, starting from $h_0 = 0$. To five decimal places,

(5.29) $$h^* \cong h_2 = -0.53213 .$$

Remark 5.1. For abstract cubic equations, condition (5.23) becomes

(5.30) $$\eta_0 \le \frac{\sqrt{b_2^2 + 6b_3} - b_2}{16b_3} ,$$

or

(5.31) $$\|B_0\| \le \frac{\sqrt{\|B_2\|^2 + 6\|B_3\|} - \|B_2\|}{16\|B_3\|} .$$

6. Solution of abstract polynomial equations by the use of the modified form of Newton's method.

The modified form of Newton's method for the solution of the abstract polynomial equation

(6.1) $$R(h) = B_n h^n + \ldots + B_2 h^2 + h + B_0 = 0 ,$$

starting from $h_0 = 0$, consists of generation of the sequence $\{h_k\}$ defined by

(6.2) $$h_0 = 0 , \quad h_{k+1} = h_k - [R'(0)]^{-1} R(h_k) ,$$

$k = 0, 1, 2, \ldots$.

By (5.6), (6.2) may be written as

(6.3) $$h_0 = 0 , \quad h_{k+1} = h_k - R(h_k) ,$$

or

(6.4) $$h_0 = 0 , \quad h_{k+1} = F(h_k) ,$$

$k = 0, 1, 2, \ldots$, where

$$\Gamma(h) = -(B_n h^n + \ldots + B_2 h^2 + B_0) . \tag{6.5}$$

Remark 6.1. The modified form of Newton's method for the solution of the abstract polynomial equation (6.1) is identical to the method of successive substitutions (4.5).

This follows immediately by comparison of (6.5) with (4.4). Consequently, an alternative condition for the convergence of the method of successive substitutions is the following theorem [14] on the convergence of the modified form of Newton's method.

Theorem 6.1 (Kantorovič-Tapia).

$$\|R''(h)\| \leq K \tag{6.6}$$

for $\bar{U}(r)$, and that

$$K\eta_0 \leq \tfrac{1}{2} . \tag{6.7}$$

If

$$r \geq r_e = \frac{1 - \sqrt{1 - 2K\eta_0}}{K} , \tag{6.8}$$

then a solution h^* of (6.1) exists in $\bar{U}(r_e)$, and the sequence $\{h_k\}$ defined by (6.4) converges to h^*, with

$$\begin{cases} \|h^* - h_k\| \leq \dfrac{\theta_k}{K}\left(1 - \sqrt{1 - 2K\eta_0}\right)^{2^k} , \\ \theta_k = \tfrac{1}{2}\left(1 + \dfrac{\beta_k}{1 - \sqrt{1 - 2K\eta_0}}\right)\theta_{k-1}^2, \ \theta_0 = 1 , \\ \beta_k = \tfrac{1}{2}\beta_{k-1}^2 + K\eta_0, \ \beta_1 = 0 . \end{cases} \tag{6.9}$$

$k = 1, 2, 3, \ldots$. If

$$r \geq r_u = \frac{1 - \sqrt{1 - 2K\eta_0}}{K} , \tag{6.10}$$

then h^* is unique in $U(r_u)$.

It is evident that the hypotheses of Theorem 6.1 are satisfied if equation (6.19) has positive solutions r_e, r_u such that

(6.11) $r_e \geq 2\eta_0 \geq r_u$.

We may then take

(6.12) $K_e = f''(r_e)$

in (6.9), and

(6.13) $K_u = f''(r_u)$

in (6.10).

Theorem 6.2. If

(6.14) $n(n-1)2^{n-1}b_n\eta_0^{n-1} + \ldots + 4b_2\eta_0 \leq 1$,

then solutions r_e, r_u of equation (6.14) exist and satisfy (6.11). Consequently, a solution h^* of equation (6.1) exists in $\overline{U}(r_e)$, and is unique in $U(r_u)$. Furthermore, the sequence $\{h_k\}$ defined by (6.4) converges to h^*, with

(6.15) $$\begin{cases} \|h^* - h_k\| \leq \dfrac{\theta_k}{f''(r_e)}\left(1 - \sqrt{1 - 2f''(r_e)\eta_0}\right)^{k+1}, \\ \theta_k = \dfrac{1}{2}\left(1 + \dfrac{\beta_k}{\sqrt{1 - 2f''(r_e)\eta_0}}\right)\theta_{k-1}, \quad \theta_0 = 1, \\ \beta_k = \dfrac{1}{2}\beta_{k-1}^2 + f''(r_e)\eta_0, \quad \beta_1 = 0 \end{cases}$$

$k = 1, 2, 3, \ldots$.

Proof. The condition (6.14) implies that the <u>indicial polynomial</u>

(6.16) $\chi(r) = n(n-1)b_n r^n + \ldots + 2b_2 r^2 - 2r + 2\eta_0$

is negative at $r = 2\eta_0$, which means that positive solutions r_e, r_u of

(6.17) $\chi(r) = 0$

... equation (6.19) exist, since $\chi(0) = 0$ and $\chi($ ... Furthermore, (6.11) is satisfied, so the conclusions of the ... follow ... Theorem 6.1.

Remark 6.2. It follows that (6.11) (and also (6.24)) will be satisfied for ... sufficiently small, say $\eta_0 < \eta$. This means that the solution $h = h_0$ of equation (6.1) depends continuously on B_0 in the neighborhood of $B_0 = 0$, ... $h^* = 0$ satisfies the homogeneous equation

$$B_n h^n + \ldots + B_2 h + h = 0 , \tag{6.18}$$

and, for any $\epsilon > 0$, a solution $h = h^*$ of (6.1) exists such that

$$\|h^*\| \leq \epsilon \tag{6.19}$$

provided

$$\|B_0\| \leq \eta = \frac{1}{2} . \tag{6.20}$$

As noted above, (4.19) and (5.19) provide alternative conditions for the convergence of the method (4.5) of successive substitutions. Actually, (5.19) ... written in the form

$$N_n(r) \equiv \frac{n(n-1)}{2} b_n r^n + \ldots + b_2 r^2 - r + \eta_0 = 0 , \tag{6.21}$$

all of the integral multipliers of the coefficients are divisible by 2. Let

$$S_n(r) \equiv n b_n r^n + \ldots + 2 b_2 r^2 - r + \eta_0 , \tag{6.22}$$

... that

$$S_n(r) \leq N_n(r) , \quad r \geq 0 , \tag{6.23}$$

... only for $n = 2, 3$, but for $n \geq 4$, it is possible that $N_n(r) < S_n(r)$.

... $$N_n(r) - S_n(r) = \frac{n(n-3)}{2} b_n r^n + \ldots + a_4 r^4 - r_2$$

... positive for r sufficiently large.

Remark 6.2. Condition (6.19) gives better estimates than (4.19) of the domain of the method of successive substitutions for quadratic and cubic equations. If $b_2 = 0$, then the situation is reversed for equations of degree greater than two.

7. Finite systems of polynomial equations. An important special case for the analysis given above occurs if $X = R^m$, the space of n-dimensional real vectors $x = (\xi_1, \xi_2, \ldots, \xi_m)$. One could equally well consider $X = C^m$, the space of n-dimensional complex vectors $z = (\zeta_1, \zeta_2, \ldots, \zeta_m)$, but it is usual in computational practice to embed complex problems in real spaces with twice the dimension of the complex space.

In R^m, an n-linear operator may be represented by an array

$$N = (v_{ij_1j_2\ldots j_n}) , \tag{7.1}$$

$i, j_1, j_2, \ldots, j_n = 1, 2, \ldots, m$ consisting of m^{n+1} elements. The following convention will be adopted for the formation of the (n-1)-linear operator Nx:

$$Nx = \left(\sum_{j_n=1}^{m} v_{ij_1j_2\ldots j_{n-1}j_n} \xi_{j_n} \right) \tag{7.2}$$

for $x = (\xi_1, \xi_2, \ldots, \xi_n)$.

Remark 7.1. The condition for the operator N defined by (7.1) to be symmetric is that

$$v_{ij_1j_2\ldots j_n} = v_{ij_{k_1}j_{k_2}\ldots j_{k_n}} , \tag{7.3}$$

$i, j_1, j_2, \ldots, m$, and $j_{k_1}, j_{k_2}, \ldots, j_{k_n}$ is any permutation of the indices $j_1, j_2, \ldots, j_n$, with $1 \le j_1, j_2, \ldots, j_n \le m$. Thus, the number of

distinct elements of a symmetric n-linear operator (7.1) is significantly smaller than for an arbitrary n-linear operator.

One important source of symmetric n-linear operators in R^n is indicated in the following remark.

Remark 7.2. If the operator F in R^m defined by

$$F(x) = (f_1(x), f_2(x), \ldots, f_m(x)) \quad , \tag{7.4}$$

where

$$f_i(x) = f_i(\xi_1, \xi_2, \ldots, \xi_m) \quad , \tag{7.5}$$

$i = 1, 2, \ldots, m$, is differentiable n times, then its nth (Fréchet) derivative at $x = x_0 = (\xi_1^{(0)}, \xi_2^{(0)}, \ldots, \xi_m^{(0)})$ is the symmetric n-linear operator

$$F^{(n)}(x_0) = \left(\frac{\partial^n f_i}{\partial \xi_{j_1} \partial \xi_{j_2} \cdots \partial \xi_{j_n}} \right)_{x = x_0} \quad , \tag{7.6}$$

$i, j_1, j_2, \ldots, j_n = 1, 2, \ldots, m$.

In R^m, abstract polynomial equations are evidently systems of m algebraic polynomial equations in the m unknowns $\xi_1, \xi_2, \ldots, \xi_m$. The systems can arise directly in applications, or be approximations to equations involving operators $F(x)$ in R^n which have power series expansions at $x = x_0$. Another source of finite polynomial systems is the discretization of polynomial operator equations in infinite-dimensional spaces. This is usually done in the case of differential equations by using finite differences as approximations to derivatives. For integral equations, corresponding finite systems are often obtained by the use of a numerical integration rule which replaces integrals by finite sums. Finite polynomial systems may also be obtained by taking a seg-

ment of an infinite system, or by other approximation techniques applied to equations in infinite dimensional spaces.

There are many topologically equivalent ways of introducing a norm in R^m, of which

$$\|x\| = \max_{(i)} |\xi_i| \tag{7.7}$$

is perhaps the simplest from a computational standpoint. The space R^m with the norm (7.7) will be denoted by R^m_∞.

For a linear operator (matrix) $L = (\lambda_{ij})$ in R^m_∞, one has

$$\|L\| = \max_{(i)} \sum_{j=1}^{m} |\lambda_{ij}| \quad . \tag{7.8}$$

For the n-linear operator N defined by (7.1) with $n > 1$,

$$\|N\| \leq \max_{(i)} \sum_{j_1=1}^{m} \cdots \sum_{j_n=1}^{m} |\nu_{ij_1j_2\cdots j_n}| \quad , \tag{7.9}$$

but equality is not necessarily attained.

The norm in $\mathfrak{L}(R^m_\infty, R^m_\infty)$ is thus defined by (7.8). For a bilinear operator $B = (\beta_{ijk})$, one has

$$\|B\| = \sup_{\|x\|=1} \max_{(i)} \sum_{j=1}^{m} \left| \sum_{k=1}^{m} \beta_{ijk} \xi_k \right| \quad , \tag{7.10}$$

from which it follows at once that

$$\|B\| \leq \max_{(i)} \sum_{i=1}^{m} \sum_{k=1}^{m} |\beta_{ijk}| \quad , \tag{7.11}$$

which is (7.9) for $n = 2$. The general case may now be established by mathematical induction.

Bilinear operators $B = (\beta_{ijk})$ in R_∞^2 may be written in the form

$$B = \left(\begin{array}{cc|cc} \beta_{111} & \beta_{112} & \beta_{121} & \beta_{122} \\ \beta_{211} & \beta_{212} & \beta_{221} & \beta_{222} \end{array}\right) . \tag{7.12}$$

For the operator

$$B = \left(\begin{array}{cc|cc} 2 & -1 & -1 & -2 \\ 1 & 0 & 0 & 1 \end{array}\right) , \tag{7.13}$$

the estimate (7.11) gives

$$\|B\| \leq 6 , \tag{7.14}$$

while direct application of (7.10) shows that

$$\|B\| = 4 . \tag{7.15}$$

Consequently, inequality (7.9) may be strict if $n > 1$.

In actual computation, it is very easy to program a computer to produce the numbers

$$b_k = \max_{(i)} \sum_{j_1=1}^{m} \cdots \sum_{j_k=1}^{m} |\beta^{(k)}_{i j_1 j_2 \cdots j_k}| , \tag{7.16}$$

$n = 2, 3, \ldots n$, given the multilinear operators in R_∞^m,

$$B_k = (\beta^{(k)}_{i j_1 j_2 \cdots j_k}) , \tag{7.17}$$

$k = 2, 3, \ldots, n$. Thus, the construction of scalar majorant polynomials can be automated. It is not essential to obtain representations of the multilinear operators entering into the equation in the form (7.17) in order to evaluate the bounds (7.16). If one is given the system

$$p_i(\xi_1, \xi_2, \ldots, \xi_m) = 0 , \qquad i = 1, 2, \ldots, m , \tag{7.18}$$

of polynomial equations of degree n in R_∞^m, then one may find the coefficients $d^{(k)}_{ij}$ of the terms of degree k in the ith equation, $i = 1, 2, \ldots, m$, and $j = 1, 2, \ldots, q_{ik}$, where q_{ik} is the number of such terms. Then,

A particular quadratic system of equations of importance in the study of matrices is the characteristic value-vector equation. Given an $m \times m$ matrix $A = (a_{ij})$, a solution $x = (\xi_1, \xi_2, \ldots, \xi_m, \lambda)$ is sought for the system

$$(7.25) \qquad \begin{cases} \sum_{j=1}^{m} a_{ij}\xi_j - \lambda\xi_i = 0 , & i = 1, 2, \ldots, m , \\ \sum_{j=1}^{m} |\xi_j|^2 - 1 = 0 . \end{cases}$$

8. Polynomial differential and integral equations. Many of the integral and differential equations of interest in applications are of polynomial type [12]. This is true, for example, of almost all of the equations listed in Chapter 1 of the book by H. T. Davis [4]. Davis also devotes Chapter 8 of his book [4] to second order differential equations of polynomial class. In the case of ordinary differential equations, the famous Riccati equation,

$$(8.1) \qquad \frac{dy}{dx} + Q(x)y + R(x)y^2 = S(x) , \qquad y(0) = c ,$$

is quadratic. If the coefficient functions Q, R, S are assumed to be continuous, then

$$(8.2) \qquad P(y) \equiv \frac{dy}{dx} + Q(x)y + R(x)y^2 - S(x)$$

may be regarded as an operator from the space $C'[0, X]$ of continuously differentiable functions on $0 \le x \le X$ into the space $C[0, X]$ of continuous real functions on the same interval.

The transformation of (8.1) into the standard form (2.17) may be carried out very simply. Choose $y_0 = y_0(x)$ in $C'[0, X]$ such that

(8.3) $$y_0(0) = c \quad ,$$

and set

(8.4) $$y(x) \quad y_0(x) + h(x) \quad .$$

The equation for the new unknown function $h(x)$ is

(8.5) $$\begin{cases} R(x)h^2 + \dfrac{dh}{dt} + [Q(x) + 2R(x)y_0(x)]h + \\ \quad + \{\dfrac{dy_0}{dx} + Q(x)y_0 + P(x)y_0^2 - S(x)\} = 0 \ , \\ \quad h(0) = 0 \ . \end{cases}$$

This is equivalent to the Taylor identity (2.25). In order for the polynomial operator (8.2) to be regular at $y = y_0$, one must be able to invert the linear differential operator

(8.6) $$P'(y_0) = \frac{d}{dx} + [Q(x) + 2R(x)y_0(x)]I$$

with the homogeneous initial condition. The inverse of (8.6) is the linear integral transform with kernel

(8.7) $$K(x, t) = \exp\{\mu(t) - \mu(x)\} \quad ,$$

where

(8.8) $$\mu(s) = \int_0^s [Q(u) - 2R(u)y_0(u)]du \quad .$$

Thus, (8.5) is equivalent to the nonlinear Volterra integral equation

(8.9) $$0 = \int_0^x K(x, t)h^2(t)\,dt + h(x) + g(x) \ , \quad 0 \le x \le X \ ,$$

where

(8.10) $$g(x) = \int_0^x K(x, t)\{\frac{dy_0(t)}{dt} + Q(t)y_0(t) + R(t)y_0^2(t) - S(t)\}\,dt \quad .$$

Equation (8.9) may be considered to be posed in the Banach space $C_0'[0, X]$ of continuously differentiable functions on $0 \leq x \leq X$ which vanish at $x = 0$. A suitable norm in this space is

$$\|f\| = \max_{[0, X]} \{ |f(x)| + |f'(x)| \} \quad . \tag{8.11}$$

It is evident from (8.7), (8.8), and (8.10) that (5.25) (or (6.14) for $n = 2$) will be satisfied if X is sufficiently small.

Higher degree polynomial integral equations of Volterra type have been studied by Lalesco [9]. In the case of boundary-value problems, a technique similar to the above yields polynomial integral equations of Fredholm type by use of the appropriate Green's function. Such algebraic integral equations have received the attention of E. Schmidt [15]. The treatment of polynomial partial differential equations may be carried out in an analogous way. For example, the two-dimensional Navier-Stokes equations [9],

$$\begin{cases} \dfrac{\partial u}{\partial t} + u\dfrac{\partial u}{\partial x} + v\dfrac{\partial u}{\partial y} = X - \dfrac{1}{\rho}\dfrac{\partial p}{\partial x} + \dfrac{\mu}{\rho}\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}\right), \\ \dfrac{\partial v}{\partial t} + u\dfrac{\partial v}{\partial x} + v\dfrac{\partial v}{\partial y} = Y - \dfrac{1}{\rho}\dfrac{\partial p}{\partial y} + \dfrac{\mu}{\rho}\left(\dfrac{\partial^2 v}{\partial x^2} + \dfrac{\partial^2 v}{\partial y^2}\right), \\ \dfrac{\partial \rho}{\partial t} = \dfrac{\partial(\rho u)}{\partial x} + \dfrac{\partial(\rho v)}{\partial y} \quad , \end{cases} \tag{8.12}$$

form a quadratic system in the incompressible case (ρ = constant). Together with (8.12), one assumes that the initial values $u(x, y, 0)$, $v(x, y, 0)$ are known, and on the boundary ∂G of some region G, $v(x, y, t)$ and $u(x, y, t)$ are specified. The body forces X, Y and viscosity μ are also given for (8.12) to be well posed.

Polynomial integral equations are often obtained by transformation of differential equations, as in the example of the Riccati equation. They also arise directly in applications, an example being the equation of Chandrasekhar

$$H(\mu) = 1 + \mu H(\mu) \int_0^1 \frac{\Psi(\mu)}{\mu+\mu'} H(\mu')\, d\mu' \tag{8.13}$$

which occurs in the mathematical theory of radiative transfer [3]. For $H(\mu) \in C[0,1]$, the space of continuous functions on $0 \le \mu \le 1$, condition (5.25) will be satisfied if

$$\max_{[0,1]} \int_0^1 \frac{\mu\Psi(\mu')}{\mu+\mu'}\, d\mu' \le \frac{1}{2} \quad , \tag{8.14}$$

as $\Psi(\mu) \ge 0$ in the problems considered.

REFERENCES

1. Banach, S., Sur les opérations dans les ensembles abstraits et leur applications aux equations integrales. Fund. Math. 3 (1922), 133-181.

2. Caccioppoli, R., Sugli elementi uniti delle transformazioni funzionali: un'osservazione sui problemi di valori ai limiti. Atti Accad. Naz. Lincei Mem. Cl. Sci. Fis. Mat. Natur. Sez I 6 (1931), 498-502.

3. Chandrasekhar, S., Radiative transfer. Dover, New York, 1960.

4. Davis, H. T., Introduction to nonlinear differential and integral equations. Dover, New York, 1962.

5. Dickson, L. E., New first course in the theory of equations. John Wiley & Sons, New York, 1939.

6. Graves, L. M., Riemann integration and Taylor's theorm in general analysis. Trans. Amer. Math. Soc. 29 (1927), 163-177.

7. Hille, E. and Phillips, R. S., Functional analysis and semi-groups. Amer. Math. Soc. Colloquium Pubs. Vol. 31. Revised Ed. American Math. Society, Providence, R. I., 1957.

8. Kantorovič, L. V., Functional analysis and applied mathematics. Uspehi Mat. Nauk 3 (1948), 89-185 (Russian). Tr. by C. D. Benster as National Bureau of Standards Report No. 1509. Washington, 1952.

9. Lalesco, T., Introduction à la théorie des equations intégrales. Paris, 1912.

10. Ostrowski, A. M., Newton's method in Banach spaces. Proceedings of the 1970 Oberwohlfach Conference on the Numerical Solution of Differential Equations, ed. by B. Brosowski and L. Collatz. To appear.

11. Rall, R. B., Quadratic equations in Banach spaces. Rend. Circ. Mat. Palermo 10 (1961), 314-332.

12. Rall, L. B., Solution of abstract polynomial equations by iterative methods. MRC Technical Summary Report #892, University of Wisconsin, Madison, 1968.

13. Rall, L. B., Computational solution of nonlinear operator equations. John Wiley & Sons, New York, 1969.

14. Rall, L. B., and Tapia, R. A., The Kantorovich theorem and error estimates for Newton's method. MRC Technical Summary Report #1043, University of Wisconsin, Madison, 1970.

15. Schmidt, E., Zur Theorie der linearen und nichtlinearen Integralgleichungen III. Math. Ann 65 (1908), 370-399.

ROUNDING

J. M. Yohe
Mathematics Research Center
University of Wisconsin
Madison, Wisconsin

1. Introduction.

There has been a great deal of work done in the areas of rounding, floating point arithmetic, and approximation of real numbers by computer representable numbers; it might seem as though we were beating a dead horse. However, to our knowledge, no manufacturer yet builds a computer which performs these functions properly -- at least, not by our definition.

In this paper, we sketch our definition of "proper" floating point hardware, indicate how it can be implemented, present the appropriate formulas for a priori error analysis based on this hardware design, and survey some of the basic applications of this arithmetic. Much of the material in this paper is treated in greater detail in [3], [6], [7], and [8]. The a priori error bounds and the extension of [6] to radix complement and sign-magnitude arithmetic, while easily deduced from [6], are not presented explicitly elsewhere. A thorough discussion of floating point arithmetic, including some of the ideas presented here, can be found in Knutch [1]; however, he does not deal with directed roundings, which we feel are essential to proper operation of a computer.

Throughout this paper, we will assume that our computer operates in the base β number system. A floating point number is a pair (E,F), where E is an m-digit signed exponent (power of β) and F is an n-digit signed fraction. Since the size and particular representation of E have no bearing on accuracy apart from limiting the size of the largest and smallest machine numbers, we will not concern ourselves with the exponent here. In all subsequent remarks which deal with number representations, it is to be understood that the statements are true only within the range permitted by the exponent size.

The floating point number (E,F) represents the number

$$\beta^E \times F, \text{ where}$$

$$F = \pm \sum_{i=1}^{n} a_i \beta^{-i}, \quad 0 \leq a_i < \beta. \tag{1.1}$$

Sponsored by the United States Army under Contract No.: DA-31-124-ARO-D-462.

It is clear from (1.1) that any number which can be expressed as an n-digit base β fraction times an m-digit power of β is a machine representable number, and no other numbers are representable. Since m and n are fixed, only finitely many different real numbers are representable. Any non-representable number r must be approximated by a machine number; if m_1 and m_2 are the two consecutive machine numbers such that $m_1 < r < m_2$, and r is approximated by either m_1 or m_2, then it is clear that the approximation is subject to an error of the order of β^{-n}. This is sometimes referred to as the basic machine precision.

A floating point number is said to be <u>normalized</u> if $a_1 \neq 0$; we will assume that all floating point numbers are normalized, since maximum accuracy is maintained by use of normalized numbers. In this terminology, zero is a special case; we will assume that it is expressed by a zero fraction and the smallest possible exponent; we will admit zero as an exception to the rule that all numbers must be normalized.

2. Roundings and Directed Roundings:

An axiomatic approach to computational rounding has been given by Kulisch in [2]. For the sake of completeness, we sketch some of the points of his theory here. We do not state the theory in full generality; those interested in further information along these lines should consult [2].

Let $\mathbb{R}$ be the real number system, and let $\mathbb{M}$ be the set of machine representable numbers. A mapping $\square : \mathbb{R} \to \mathbb{M}$ is said to be a <u>rounding</u> if, for all $a, b \in \mathbb{R}$ we have

$$\square a \leq \square b \text{ whenever } a \leq b.$$

A rounding is called <u>optimal</u> if for all $a \in \mathbb{M}$, $\square a = a$. In practice, this must be true for any reasonable representation of a, which must certainly include any representation the computer might manufacture during an intermediate stage of an arithmetic operation. The definition of optimal rounding implies that if $a \in \mathbb{R}$ and m_1, m_2 are consecutive members of $\mathbb{M}$ with $m_1 < a < m_2$, and if $\square : \mathbb{R} \to \mathbb{M}$ is an optimal rounding, then either $\square a = m_1$ or $\square a = m_2$.

A rounding is <u>downward directed</u> (<u>upward directed</u>) if, for all $a \in \mathbb{R}$, we have $\square a \leq a$ ($\square a \geq a$). A rounding is <u>symmetric</u> if $\square a = -\square(-a)$. If $\square$ us a rounding, a,b are machine numbers, and * is an arithmetic operation, then by $a \boxed{*} b$ we will mean $\square(a*b)$.

By Theorem 1 of [2], optimal directed roundings are unique. We denote the optimal upward directed rounding by Δ, the optimal downward directed rounding by ∇, and the symmetric rounding which takes each real number to the closest machine number (rounding to the next machine number whose magnitude is larger if there is a tie) by $\circ$.

3. Floating point hardware design.

By our definition, proper hardware design would enable the computer to perform any of the roundings Δ, ∇, and $\circ$ at the user's option. The rounding $\circ$ is most frequently used, since it produces maximal accuracy. However, the roundings Δ and ∇ are used in implementation of interval arithmetic, for example, and in certain other situations; we will discuss applications briefly in Section 5. We will refer to floating point arithmetic which provides for all three of these roundings as "Best Possible" floating point arithmetic. The following is a brief sketch of the theory presented in [6].

What information does our computer need in order to round a real number properly? It clearly needs the first n digits of the appropriate base β fraction. Moreover, in order to be able to round to the nearest machine number (by our definition of such rounding) it needs the $n+1^{\underline{st}}$ digit of the fraction. Finally, in order to obtain a correct upward or downward directed rounding, it needs an indicator to tell us whether there are any nonzero digits in the remainder of the fraction.

The result of an arithmetic operations combining two machine numbers is not, in general, a machine number; we must usually approximate the answer. In order to assure ourselves that our computer has all of the above information at the conclusion of an arithmetic operation, we must design it to preserve even more information during the execution of the operation. We will illustrate this by means of a floating point decimal representation which uses a three digit fraction and sign-magnitude representation for negative numbers. We will confine our discussion to addition, sin[illegible] multiplication and division

We will also need two guard digits at the right-hand end of the register to preserve information which is shifted out of the right-hand end of the three-digit fraction. These guard digits are appended to the three fractional digits to form a five digit fraction; all five digits participate fully in the addition. The initial value of the guard digits is, of course, zero; in (β-1)'s complement arithmetic, zero is expressed as zero if the number is positive and as (β-1) if the number is negative. The need for one guard digit is self-evident; the need for two is illustrated by the following problem:

$$.100 \times 10^{0}$$

$$-.995 \times 10^{-2}$$

This should be computed as follows in sign-magnitude arithmetic:

$$\begin{array}{r} .10000 \\ -.00995 \\ \hline =.09005 \end{array}$$

Normalization now yields $.90050 \times 10^{-1}$; the former second guard digit is now the first guard digit, and is necessary for proper rounding.

One might wonder whether an unlimited number of guard digits would be necessary. The following theorem shows that two guard digits are always sufficient to preserve maximal accuracy:

Theorem 3.1: If more than one position of left shift is required to normalize the result of an addition, then at most one position of right shift was required to equalize the exponents.

The proof of Theorem 3.1 is given in [6].

The two guard digits are denoted by GG in Figure 1

The final item of information needed is an indicator to show whether any nonzero digits were shifted off during equalization of the exponents. This indicator can be a single binary digit, and is denoted by I in Figure 1.

Figure 1. The structure of the accumulator

Explicit algorithms for Best Possible floating point arithmetic are presented in [6]. These algorithms are stated for $(\beta-1)$'s complement arithmetic, but they are, in fact, perfectly general. In order to use them for sign-magnitude or β's complement arithmetic we need only set the guard digits to zero upon loading a number into the C.P.U., regardless of the sign of the number. A person wanting to implement Best Possible floating point arithmetic on a sign-magnitude or β's complement machine might think that modification of the algorithms would be necessary, because the proof of the algorithm is given for $(\beta-1)$'s complement arithmetic. We sketch the reasoning for the other forms of arithmetic below.

The algorithm in [6] requires that the number with the smaller magnitude be made positive before addition is begun. This number is then placed in the accumulator and shifted right the requisite number of places to equalize the exponents; if any nonzero digits are shifted out of the low-order guard digit, I is set to 1.

The sum is now formed; it is negative unless it is identically zero. However, if I is nonzero, it represents a positive correction, since the number which caused I to be set to 1 was positive. Thus the correct result lies between the number we have generated and the next larger number representable on the machine. The next larger number is clearly the negative number whose magnitude is next smaller than the magnitude of the number we have generated.

In order to proceed with the algorithm, we need to obtain a lower bound for the magnitude of the result; the magnitude of what we have obtained is an upper bound. Consequently, if I is nonzero, indicating that the lower and upper bounds are not the same, we add one into the low order digit position, which decreases the magnitude of the number, and then make it positive; we can now proceed with the algorithm. (of course, we record all of these sign changes so we can apply any necessary correction at the completion of the operation).

As an example of this, let us consider the following problem:

$$.100 \times 10^{0}$$

$$-.990 \times 10^{-4}$$

In 9's complement arithmetic, the computation is as follows (the indicator I is shown as the 1 two spaces to the right of the low-order guard digit).

```
      .00009  1
  -  +.89999
 =-   .90008  1          Form sum
           1             Add 1 to result
  -   .90009  1
  +   .09990  1          Complement
```

In 10's complement arithmetic, the computation looks like this:

```
      .00009  1
  -  +.90000              Form sum
 =-   .90009  1
           1             Add 1 to result
  -   .90010  1
      .09990  1          Complement
```

In sign-magnitude arithmetic, we have

```
  -   .00009  1
     -.10000              Form sum
 =-  -.09991  1
           1             Subtract 1 from magnitude
  -  -.09990  1               (i.e., add 1 to result)
      .09990  1          Negate
```

The results of the three types of arithmetic are identical -- which they should be, since positive numbers are expressed the same way no matter which type of arithmetic we are using. The floor and ceiling values for the sum are, respectively, $.999 \times 10^{-1}$ and $.100 \times 10^{0}$, which is exactly what we would expect from any of the three methods. Hence Best Possible floating point arithmetic could be implemented on any hardware, regardless of the arithmetic scheme used.

Although we have avoided any mention of exponent overflow and underflow conditions, proper hardware design should include proper handling of out-of-range numbers. This includes an interrupt upon occurrence of the error condition, together with a complete set of indicators to tell the user exactly what went wrong. Details of such a scheme are given in [6].

One further word about hardware is appropriate: if the machine operates in the base $\beta \neq 10$, then the hardware ought to provide facilities for conversion between base β and 10. If the hardware is designed to do arithmetic operations and rounding in the manner described here, accurate conversions -- at least from base 10 to base β -- should also be relatively easy to incorporate. This is discussed in detail in [8], and we will explore it no further here.

4. A priori error analysis:

The standard reference work on *a priori* error estimates for floating point arithmetic is Wilkinson's 1963 paper [5]. For multiplication and division, Wilkinson's error estimates are as natural as can be expected; however, in the case of addition (and subtraction) the capriciousness of computer designers made it necessary to produce a rather unnatural and somewhat intractable error estimate in order to reflect the realities of the situation. Here, we will see that Best Possible floating point arithmetic yields a more natural and more tractable error bound for addition than can be hoped for if the computer produces less than optimal accuracy.

Throughout this section, as in the rest of the paper, we assume that neither overflow nor underflow occurs during arithmetic operations. Overflow is almost invariably a fatal (to the computation) error; underflow can often be tolerated, and calculation can proceed with a zero replacing the underflowed quantity. (All the same, underflow should at least optionally trigger an error indicator or interrupt so that it can be detected; the absence of automatic error detection on underflow can lead to failure to recognize invalid computational results!) Replacing an undersized result with zero complicates the error analysis; this has been considered by Schoenfeld in [4].

If * is any of the four arithmetic operations in the real field, then by $*_M$ we will denote the machine approximation to *. The constants μ, δ, and α which appear in the following formulas are determined by the hardware design, but are essentially of the order of β^{-n}. In each formula, θ is a constant in the range $-1 \leq \theta \leq 1$ which depends on the operation and the operands; to keep notation uncluttered, we will not reflect this dependency.

Wilkinson's *a priori* formulas are as follows:

$$x \cdot_M y = (x \cdot y)\ (1+\theta\mu) \tag{4.1}$$

$$x \div_M y = (x \div y)\ (1+\mu\delta) \tag{4.2}$$

$$x +_M y = x(1+\theta\alpha) + y(1+\theta'\alpha) \tag{4.3}$$

These formulas imply that

$$|x \cdot_M y - x \cdot y| \leq |x \cdot y|\mu \tag{4.4}$$

$$|x \div_M y - x \div y| \leq |x \div y|\delta \tag{4.5}$$

$$|x +_M y - x+y| \leq \alpha(|x| + |y|) \tag{4.6}$$

Typical values for the constants u, ζ, and α are $\frac{1}{2}\beta^{1-n}$, $(\beta - \frac{1}{2})\beta^{-n}$, and $2\beta^{-n}$, depending on the design of the hardware.

If we are using Best Possible arithmetic, however, we can regard our machine M as being three machines, known as O, Δ, V, which perform the roundings O, Δ, and V (described in Section 2) respectively. For each of these machines, we have $x *_{\square} y = x \boxed{*} y$ whenever * is one of the four arithmetic operations. Moreover, we can replace (4.3) with the formula

$$x \boxed{+} y = (x + y)(1 + \theta\alpha) \qquad (4.3')$$

which implies that

$$|x \boxed{+} y - x + y| \leq |x + y|\alpha \qquad (4.6')$$

These estimates are clearly more natural and perhaps more aesthetically pleasing than (4.3) and (4.6). Moreover, in the case of rounding, we have $u = \zeta = \alpha = \frac{1}{2}\beta^{-n}$, while in the cases of upper and lower bounds, $u = \sigma = \lambda = \beta^{-n}$.

Explicitly, we have

$$x \circledast y = (x * y)(1 + \theta\beta^{-n}), \quad -\tfrac{1}{2} \leq \theta \leq \tfrac{1}{2} \qquad (4.7)$$

$$x \mathbin{⍋} y = x * y + \theta|x * y|\beta^{-n}, \quad 0 \leq \theta < 1 \qquad (4.8)$$

$$x \mathbin{⍒} y = x * y - \theta|x * y|\beta^{-n}, \quad 0 \leq \theta < 1 \qquad (4.9)$$

which implies that

$$|x * y|(1 - \tfrac{1}{2}\beta^{-n}) \leq x \circledast y \leq |x * y|(1 + \tfrac{1}{2}\beta^{-n}) \qquad (4.10)$$

$$x * y \leq x \mathbin{⍋} y < x * y + |x * y|\beta^{-n} \qquad (4.11)$$

$$x * y - |x * y|\beta^{-n} < x \mathbin{⍒} y \leq x * y \qquad (4.12)$$

Moreover, it should be noted that whenever the result of any operation is a machine representable number, then the result of the operation is exact. This, unfortunately, is not always the case with present day hardware.

As an extreme example, let us consider the case of a machine with a ten digit fraction, operating in the decimal number system; let us assume that $\epsilon = \frac{1}{2}\, 10^{-10}$. Then, if we have the following addition

$$\begin{array}{r} .1000000000 \times 10^{1} \\ \underline{-.9999999999 \times 10^{0}} \\ .0000000001 \times 10^{0} = .1000000000 \times 10^{-9} \end{array}$$

the error bound obtained from (4.6) would tell us that the result is $.1000000000 \times 10^{-9} \pm .99999999995 \times 10^{-10}$, or, essentially, that we have no significance left. However, (4.6') says that the result is $.1000000000 \times 10^{-9} \pm .1000000000 \times 10^{-19}$, which is a far more optimistic bound on this particular addition!

It can be argued that these summands are probably inaccurate in the last decimal place, so that the Wilkinson bound is more realistic. Perhaps this is the case; however, that decision should be left to appropriate error analysis on the summands. The important fact here is that, using Best Possible arithmetic, the result of the above problem is computed exactly, and consequently, the less uncertainty the bound reflects, the better it is.

The necessity for such a pessimistic bound as that in (4.6) can be seen from looking at the above example as it might be computed by a computer using typical present-day design:

$$\begin{array}{r} .1000000000 \times 10^{1} \\ \underline{-.09999999999 \times 10^{1}} \\ .0000000001 \times 10^{1} = .1000000000 \times 10^{-8} \end{array}$$

Here, of course, we would have $\epsilon \sim .100000000 \times 10^{-9}$ since the machine truncates before normalizing, and consequently (4.6) yields an estimate of $.1000000000 \times 10^{-9} \pm .19999999999 \times 10^{-8}$; this is not unduly pessimistic. Of course, (4.6') does not apply to hardware of this design.

5. Applications

We will mention a few of the applications of Best Possible floating point arithmetic. The rounding ○ has applications in almost every computation using floating point arithmetic. It is this rounding that we expect to get, and (usually erroneously) assume we do get, from a piece of equipment costing several million dollars.

The roundings $\triangle$ and $\triangledown$, while not provided with any production computer we know of, also have important applications.

Perhaps the most obvious application of these roundings is in the implementation of interval arithmetic. Hardware designed to produce these roundings would render the programming of an interval arithmetic package nearly trivial, and would enable interval operations to be executed in one tenth to one fifth of the time normally required to execute them with simulated floating point arithmetic (simulation is usually necessary if we are to be able to produce the tightest possible bounds). The formula for addition of two intervals, for example, is

$$[a,b] + [c,d] = [a + c,\ b + d] .$$

If we assume that a,b,c, and d are machine representable numbers and denote the computer approximation to [a,b] + [c,d] by $[a,b] \diamondsuit [c,d]$, then the above formula translates as follows:

$$[a,b] \diamondsuit [c,d] = [a \triangledown c,\ b \triangle d].$$

Evaluation of this formula on a computer equipped with directed rounding takes just twice as long as evaluating the sum of two floating point numbers.

An interval arithmetic package for the UNIVAC 1108, using simulated floating point arithmetic as described in [6], is detailed in [3].

Another consequence of directed roundings is that upper and lower bounds for sequences of machine operations are much more easily and accurately computed, both a priori and during computation, than is possible with conventional rounding. This enables one to combine a priori analysis with computational considerations to produce rigorous bounds for relative error in evaluation of mathematical functions. In [7], it is proved that on a binary computer with optimal directed rounding, the square root of a machine representable number can be calculated exactly if it is machine representable, and bracketed by two consecutive machine numbers if it is not machine representable; this is accomplished without using interval arithmetic.

Perhaps the main point of this paper can be summed up very simply: floating point arithmetic can be this good, and it can be no better. Users of computing equipment should not have to settle for less.

REFERENCES

1. Donald E. Knuth, The Art of Computer Programming, Volume 2, Semi-numerical Algorithms, Addison-Wesley Publishing Co., Reading, Mass. 1969.

2. U. Kulisch, An axiomatic approach to rounded computation, Technical Summary Report #1020, Mathematics Research Center, University of Wisconsin, November, 1969.

3. T. D. Ladner and J. M. Yohe, An interval arithmetic package for the UNIVAC 1108, Technical Summary Report #1055, Mathematics Research Center, University of Wisconsin, May, 1970.

4. Lowell Schoenfeld, Floating Point Error Estimates, Chapter IX of Technical Summary Report #721, Mathematics Research Center, University of Wisconsin, August, 1967.

5. J. H. Wilkinson, Rounding Errors in Algebraic Processes, National Physical Laboratory Notes on Applied Science, No. 32, Her Majesty's Stationery Office, London, 1963.

6. J. M. Yohe, Best Possible Floating Point Arithmetic, Technical Summary Report #1054, Mathematics Research Center, University of Wisconsin, March, 1970.

7. _____, Rigorous Bounds on Computed Approximations to Square Roots and Cube Roots, Technical Summary Report #1088, Mathematics Research Center, University of Wisconsin, September, 1970.

8. _____, Accurate Conversion Between Number Bases, Technical Summary Report #1109, Mathematics Research Center, University of Wisconsin, October 1970.

NUMERICAL SPECTRA AND APPLICATIONS TO COMPUTATIONAL METHODS

M. Z. Nashed and K. Orlov
Mathematics Research Center
University of Wisconsin
Madison, Wisconsin

ABSTRACT. This paper develops a new technique for computations based on the theory of numerical spectra which results in a considerable reduction in the required arithmetic operations and over-all computing time. The essence of this spectral method is an arithmetization of the elements and algebraic operations occurring in a computational problem. The method consists of forming from each element entering the computation (for instance, a polynomial of an algebraic equation, a matrix, an interval, etc) one number called the spectrum of that element. Calculations with such numbers-spectra are done in the same way as with ordinary numbers. Each of the spectra occupies one cell of the digital computer. After all the necessary operations have been performed on the spectra, the resulting spectrum gives the solution of the problem in spectral form. By applying the same rules as in forming the spectrum, but inversely, one obtains a sequence of numbers which gives the solution.

The purpose of this paper is two-fold: First, a systematic and unifying theory of spectra and pseudospectra is presented, with special attention being given to internal operations within a given spectrum and binary arithmetic operations on spectra. Second, applications of numerical spectral analysis are given to computations with recursive relations, difference and differential equations, interval arithmetic, solution of polynomial equations by Graeffe's and Bernoulli's methods, and some computational methods of linear algebra.

1. INTRODUCTION. The necessity to deal with complex problems in mathematics had led to the introduction of mathematical entities that are more complicated than the real numbers. Examples of such entities include vectors, matrices, tensors, polynomials, etc. They are composed of, or related to , numbers in a certain way but they are not more numbers. Operations with such entities are more complicated than arithmetic operations with numbers. In an effort to simplify and to perform automatically such operations, various arithmetics and routines have been developed. In the present paper, we develop a theory of numerical spectra, to be defined in Section 2, and apply it to computational arithmetic and numerical methods. At the outset, we should like to point out that the terms "spectrum" and "spectral analysis" as used in this paper have no connection with their connotations in functional analysis [9].

The notation of a mathematical spectrum is due to Petrovitch [20]. The idea of the simplest numerical spectrum is to establish a certain one-to-one correspondence between a set of elements and a set of positive integers, methods was made by Orloff [14], [15].

In Section 2, the basic theory of numerical spectra is presented, with particular attention being given to internal operations within a spectrum, binary arithmetic operations with spectra, and operations of choice. The formation of spectra of various elements occurring in numerical problems are also discussed. In Section 8, pseudospectra are introduced and applied to solving initial-value problems of differential equations. In Sections 3-7, new applications of spectra are given to arithmetic, polynomials, recursive calculations and difference equations, interval arithmetic, and solutions of polynomial equations. The applications of spectra to Graeffe's and Bernoulli's methods of solving polynomial equations and to computational methods of linear algebra are refinement of earlier results [14], [17], [18], [10].

Examples accompany most of the spectral techniques introduced in the present paper. These examples demonstrate the computational procedures in the context of spectral analysis, and show the reduction in the number of operations, the simplicity and over-all reduction of time in comparison with ordinary methods. For these advantages it is felt that the spectral method should be of considerable interest for computers. The spectral method may also play a role in further developments of digital computers.

2. BASIC THEORY OF NUMERICAL SPECTRA.

2.1 THE SPECTRA OF A SEQUENCE OF POSITIVE INTEGERS. Consider the finite sequence of positive integers

$$5,\ 13,\ 8,\ 28 \qquad (2.1)$$

These integers have different numbers of digits. Before forming the spectrum of this sequence, these numbers must be transformed into so-called *spectral numbers* (spectral integers), that is, integers having the same number of digits. The necessity to preserve the value of each number of the sequence (2.1), and at the same time to transform them into spectral numbers leads to the completion of such numbers by zeros on their left. Thus the sequences

$$05,\ 13,\ 08,\ 28$$
$$005,013,008,028$$

are sequences of spectral integers. The numbers of digits in every spectral integer in the same sequence is called the uniform spectral rhythm or briefly the *rhythm* h. The rhythm of the former sequence is two while the rhythm of the latter is three.

The *ordinary spectrum* or briefly the *spectrum* of a sequence of positive integers is the *number* obtained by writing the spectral numbers consecutively. For example, the numbers $S_1 = 05130828$ and $S_2 = 005013008028$ are two spectra of the same sequence (2.1), with rhythms 2 and 3 respectively.

The spectrum can be partitioned into sections, each section containing only one spectral number. Thus the spectra S_1 and S_2 are written

$$S_1 = 05|13|08|28, \qquad S_2 = 005|013|008|028$$

The sections are enumerated from left to right. The zeros on the left of the spectrum are ordinarily omitted. Thus the length of each section of the spectrum is the same with the possible exception of the first section which can have fewer digits. Note that the rhythm h = 1 is not compatible with the sequence (2.1). The individuality of each of the numbers in (2.1) definitely is not lost and can be restored if necessary by the operation of cutting the spectrum.

The ordinary spectrum is not the only spectrum used in computation. Another kind of spectrum, called inverse spectrum, is obtained in the following way. The numbers of the sequence are arranged in opposite order and afterwards the ordinary spectrum is formed. For example, the inverse spectrum of the sequence (2.1) with rhythm h = 2 is the number $\Sigma = 28081305$. The unit spectrum is defined to be the spectrum composed of a spectral number one in each of the sections. For example, the unit spectrum of four terms with h = 2 is 1010101.

2.2 THE SPECTRUM OF A SEQUENCE OF INTEGERS OF DIFFERENT SIGNS.

A sequence of integers with different signs, for instance the sequence

$$12, \quad -27, \quad 48, \quad 8, \quad -13 \qquad (2.2)$$

can be decomposed into two sequences. The first is composed from the positive terms of the sequence (2.2) and zeros in the places of the negative numbers, that is,

$$12, \quad 0, \quad 48, \quad 8, \quad 0 \qquad (2.2')$$

The second sequence is formed from the absolute values of the negative terms with zeros replacing the positive terms in the sequence, that is,

$$0, \quad 27, \quad 0, \quad 0, \quad 13 \qquad (2.2'')$$

Let S^+ and S^- denote the spectra of the sequences (2.2') and (2.2'') with a compatible common rhythm h. S^+ and S^- are called respectively the positive and negative spectrum of the sequence (2.2). The difference $S = S^+ - S^-$ is defined to be the spectrum of the sequence (2.2) of positive and negative integers. The compatible rhythm h with a sequence of positive and negative integers must satisfy the following inequality

$$\max |a_i| < \frac{10^h}{2}$$

where the a_i's are the terms of the sequence. In the example, the lowest compatible rhythm is h = 2 and the spectrum formed with this rhythm is

$$S = 11|73|48|07|87 \qquad (2.3)$$

The sections of the spectrum are called <u>big sections</u> if they begin with big digits (5 - 9) and <u>small sections</u> if they begin with small digits (0 - 4). The beginning digit itself is called the <u>characteristic digit</u> of the section.

Now it is necessary to give the solution of the inverse problem of the forming the spectrum, i.e., to recover the sequence from the spectrum. For this purpose we introduce the notations of nominal, corrected and effective value of a section of a spectrum. The <u>nominal</u> value is just the number written in this section of the spectrum. The <u>corrected</u> value of a section is equal to its nominal value if the characteristic digit of the following section is a small digit and is one greater than the nominal value if this characteristic digit is a big one. The corrected value of the last section is the same as the nominal value. The <u>effective</u> value of a small section is equal to the corrected value of this section; the effective value of a big section is a negative number which is the difference between the corrected value of this section and 10^h. It follows readily that the <u>effective values of the sections are the terms of the sequence</u>. In the previous example, the nominal values of the sections are

11, 73, 48, 07, 87,

the corrected values are

12, 73, 48, 08, 87,

and the effective values are

12, -27, 48, 8, -13.

If a spectrum has an odd number of sections, the effective value of the middle section is called the <u>middle part</u> of the spectrum and is denoted by M(S). In the case of a spectrum with even numbers of sections, we have two middle sections denoted by $M_i(S)$, i = 1,2. Obviously it is possible to form an inverse spectrum of a sequence composed from positive and negative integers. The inverse spectrum of the sequence (2.2) with the rhythm h = 2 is

$$\Sigma = -12|91|52|26|88 \qquad (2.4)$$

The ordinary (inverse) spectrum is either a positive or a negative number. In order to recover the terms of the sequence from a spectrum which is a negative number, we first obtain the effective values of the sections of the absolute value of the spectrum and then change the sign of every effective value.

2.3. OPERATIONS WITH SPECTRA. Spectra are numbers. It is possible therefore to perform the usual arithmetical operations on spectra. But there are internal operations which are applicable only to the spectrum such as the operations of cutting of the spectrum into sections, mentioned earlier. Another cutting of the spectrum into parts, which are not the sections, is ordinarily used in the case when the spectrum has too many digits, so it cannot enter into the computer as a whole. In this case the length of the parts is determined by the capacity of the computer or desk calculator. Such cutting is a purely operational one without other significance.

Another operation is the transposition of the spectrum to another rhythm. A transposition to a greater rhythm is called a dilution of the spectrum and the new spectrum is called a diluted spectrum. The dilution of a spectrum from a rhythm h to a rhythm $H > h$ is obtained by putting $H - h$ digits of 9 on the left of each big section and $H - h$ digits of 0 on the left of each small section, with the exception of the first section which is left unchanged. For example the dilution of the spectrum (2.3) to the rhythm $H = 4$ gives the spectrum

$$11|9973|0048|0007|9987 \qquad (2.5)$$

The dilution of the inverse spectrum (2.4) to $H = 3$ gives the spectrum

$$\Sigma = -12|991|952|026|988.$$

The transposition from a rhythm H to a smaller rhythm h is called condensation. If a spectrum with rhythm H has the property that every section of the spectrum begins with $H - k$ zeros or $H - k$ digits of 9, then this spectrum can be condensed to a smaller rhythm $h = H - j$, $j = 1,2,\ldots,k-1$. It can be condensed to the rhythm $h = H - k$ only if the $(k + 1)^{st}$ digit in each section is of the same kind (big or small) as the characteristic digit of this section. The condensation is effected by the deletion of the first j digits of each section. For example the spectrum

$$11|9943|0007|9987$$

can be condensed to the rhythm $h = 3$, but not to the rhythm $h = 2$, whereas the spectrum

$$11|9973|0007|9987$$

can be condensed to the rhythm h = 2.

The operation of elongation is applied directly to a sequence and consists of adjoining to the sequence in certain places new terms of zero. For example, the sequence (2.2) elongated on the second, fourth and fifth terms will be

$$12,\ 0,\ -27,\ 0,\ 0,\ 48,\ 8,\ -13.$$

Sometimes for certain purposes it is necessary to form a spectrum which is slightly different from the ordinary (or inverse) spectrum. Such kind of spectrum is called corrected spectrum and is denoted by $\bar{S}$. The most useful corrected spectrum is the spectrum obtained by subtracting from certain sections of the original spectrum twice their effective values. For example we will use the spectrum corrected in the last section or the spectrum corrected in all the even sections.

The operation of ablation of the last r sections in a spectrum of rhythm h consists of rounding off (in ordinary way) this spectrum by rh digits and then the deletion of the last rh zeros. Note that to the ablated spectrum corresponds the same sequence without its last r terms. The ablated spectrum is denoted by $\tilde{S}$. For example, the ablation of the last section of the spectrum (2.3) leads to

$$\tilde{S} = 11|43|48|08$$

Another important operation with spectra is the internal or formal rounding off. This operation consists of applying the ordinary rounding off, but only to the big sections. The operation is carried out beginning from right to left; the big sections are replaced by zeros and each spectral number on the left of a big section is increased by one. For example, the spectrum (2.3), formally rounded off will be

$$12|00|48|08|00$$

It is obvious that in this way we obtain the positive spectrum S^+ of the original sequence (2.2). We give another example of formal rounding off. The spectrum S = 0|9986|9981|9979|0041|9999|0001 belongs to the sequence 1, -13, -18, -21, 42, -1, 1 for h = 4, and the formally rounded off spectrum is 1|0000|0000|0000|0042|0000|0001.

Finally we shall mention the operation of choice. Examples of such operations were described earlier by obtaining the positive and negative spectra S^+ and S^- of the sequence (2.2), where the positive and negative terms were chosen respectively. The mostly used choice

operations besides those just mentioned are: (i) the choice of the odd terms of the sequence; (ii) the choice of the even terms of the sequence. In all the operations of choice, all the non-chosen terms are replaced by zeros. The spectra of the transformed sequences under the operations of choice (i) and (ii) are denoted by S_1 and S_2 respectively. Combined with the choice of positive and negative terms, these lead to four new operations of choice:

(1) the choice of positive terms of the sequence on odd places;
(2) the choice of negative terms of the sequence on odd places;
(3) the choice of positive terms on even places;
(4) the choice of negative terms on even places.

The corresponding spectra are denoted by S_1^+, S_1^-, S_2^+ and S_2^- respectively. Note that in a choice of negative numbers (S^-, S_1^-, S_2^-), the transformed sequence is formed by replacing each chosen negative by its absolute value, and each of the non-chosen numbers by zero; thus the four spectra designated above are spectra of sequences of negative integers.

2.4. SPECTRA OF RATIONAL NUMBERS, INTERVALS, VECTORS, MATRICES, POLYNOMIALS AND FUNCTIONS. The spectrum of a rational number a/b is defined to be the spectrum of the sequence a,b. We also define the spectrum of an interval [a:b] to be the spectrum of the sequence a,b. This latter definition enables us to apply numerical spectra to computations with interval arithmetic ([12], [22], [6]). (This dual use of the spectrum of a sequence a,b should not cause any confusion if we stress the mathematical notions we are dealing with). The spectrum of a vector is the spectrum of the sequence of its components. The row (column) spectrum of a matrix is the spectrum of all of its elements taken row by row (column by column). The decreasing spectrum S_d of a polynomial $P(x) = a_o x^n + a_1 x^{n-1} + \ldots + a_n$ is the spectrum of the sequence $a_o, a_1, \ldots, a_n$. The increasing spectrum S_i of the polynomial P is the spectrum of the sequence $a_n, \ldots, a_o$. It is obvious that the relation between these two numbers is the same as between ordinary and inverse spectra of the same sequence of numbers.

One of the most important properties of the spectra of polynomials is that the decreasing spectrum of a polynomial is the numerical value of the polynomials for $x = 10^h$:

$$S_d = P(10^h). \tag{2.6}$$

Similarly we have

$$S_i = 10^{nh} P(10^{-h}) \tag{2.7}$$

where n is the degree of the polynomial and h is the rhythm of the spectrum. The subscripts d and i will be dropped when one kind of

spectra of polynomials is used.

In many applications it is not necessary to require the spectrum to be an integer; the value $P(10^{-h})$ may be considered as a spectrum of the polynomial. Such a spectrum is called noninteger spectrum of the polynomial and is denoted by S_i^*. Thus

$$S_i^* = P\ (10^{-h}) \tag{2.8}$$

This new notion of spectrum gives a possibility of forming spectra of functions.

Let a function f possess the following properties: (1) it has the Taylor series expansion at the point a in its domain Ω; (2) the coefficients of the series are integers (+) admitting a rhythm h such that $a + 10^{-h} \in \Omega$. Then the number

$$S^* = f(a + 10^{-h}) \tag{2.9}$$

is defined to be the noninteger spectrum with rhythm h of the function f at the point a. It is obvious that such a spectrum is an infinite decimal number. Note that if the spectrum $f(a + 10^{-h})$ is known approximately, to the nh^{th} decimal, it represents the spectrum of the Taylor polynomial of degree n associated with f at the point a.

OPERATIONS OF CHOICE APPLIED TO POLYNOMIALS. The operations of choice defined for sequences can also be applied to polynomials. We consider a polynomial P(x) arranged by decreasing powers. One operation of choice leads to the polynomials $P_1(x)$ and $P_2(x)$ composed from odd (respectively even) terms of the polynomial. If n is an even number, then P_1 is an even function and P_2 is an odd function, and vice versa if n is an odd number.

Another operation of choice leads to the "positive part", denoted by $P^+(x)$, of the polynomial P(x) which is the part composed from terms with positive coefficients. Similarly the negative part, denoted by $P^-(x)$, is the polynomial composed from absolute values of the negative terms of P(x).

Consecutive applications of these two choices lead to four polynomials $P_1^+(x)$, $P_1^-(x)$, $P_2^+(x)$, $P_2^-(x)$ defined as follows:

$$P_j^+(x) = [P_j(x)]^+, \quad P_j^-(x) = [P_j(x)]^-, \quad j = 1,2.$$

Thus we may associate with each given polynomial, in addition to its complete spectrum S, the following spectra S_1, S_2, S^+, S^-, S_1^+, S_1^-, S_2^+, S_2^-. The numerical values of these spectra are

(+) This requirement is weakened in subsequent sections.

$$S = P(10^h)$$

$$S^+ = P^+(10^h), \quad S^- = P^-(10^h)$$

$$S_j^+ = P_j^+(10^h), \quad S_j^- = P_j^-(10^j), \quad j = 1, 2.$$

2.5 SPECTRA OF POLYNOMIALS IN SEVERAL VARIABLES. Let $P(x,y)$ be a polynomial in two variables with integer coefficients. We arbitrarily choose one of the variables, for instance x, to be the principal variable. Then $P(x,y)$ can be written in the form

$$P(x,y) = \sum_{j=1}^{n} a_j(y)\, x^j \tag{2.10}$$

where $a_j(y)$ is a polynomial in y with integer coefficients. Let m denote the greatest degree of the polynomials $a_j(y)$, $j = 0,1, \ldots, n$. Let h denote a compatible common rhythm for these polynomials. The decreasing spectrum of the polynomial $a_j(y)$ is given by

$$s_j = a_j(10^h), j = 0,1,2, \ldots, n. \tag{2.11}$$

Now we form the polynomial

$$Q(x) = \sum_{j=1}^{n} s_j\, x^j \tag{2.12}$$

Taking into consideration the fact that each of coefficients s_j is an integer with absolute value less than $\frac{1}{2}10^{(m+1)h}$, we can form all kinds of spectra of the polynomial $Q(x)$ with the rhythm $H = (m+1)h$ or with greater rhythm. For instance, the decreasing spectrum S of the polynomial $Q(x)$ will have the following value

$$S = Q(10^H) = \sum_{j=0}^{n} s_j\, 10^{jH} = \sum_{j=0}^{n} 10^{jH}\, a_j(10^h) = P(10^H, 10^h) \tag{2.13}$$

The last formula is obtained using (2.10) - (2.12).

This kind of spectrum is also called spectrum with double rhythm H and h. The rhythm H is usually taken to be a multiple of the rhythm h.

Thus the numerical value $P(10^H, 10^h)$ is called the decreasing spectrum in x and y of the polynomial $P(x,y)$ and is denoted by S_{dd}. The number $P(10^H, 10^{-h})$ is denoted by S_{di}^+ and is

called the spectrum decreasing in x and increasing in y. Similarly $S^{*}_{id} = P(10^{-h}, 10^{H})$ and

$$S^{*}_{ii} - P(10^{-H}, 10^{-h}) \qquad (2.14)$$

The last notion of spectrum (2.14) can be extended to functions developable in Taylor series with integer coefficients and satisfying certain conditions.

The spectra of polynomials in any finite number of variables may be obtained in the same way as for polynomials in two variables.

REMARK: A spectrum of a matrix can be considered as a spectrum with double rhythm. In this case the spectrum is partitioned by means of the rhythm H into sections corresponding to the rows (or columns) and afterwards by means of the rhythm h into subsections corresponding to the elements. If H = mh, where m is the number of rows (or columns), this different appraoch will have no effect on the numerical value of the spectrum itself. But if H ≠ mh, then the spectra formed by means of only h, or by means of H and h, will have different numerical values.

REMARK: At the first glance the condition that the terms of the sequence must be integers appears to be a severe restriction on the use of the spectra. Essentially this is not true, because if the sequence is formed from finite decimal numbers, they can be transformed for purposes of numerical spectra into integers (by multiplication by 10^{k}, for some sufficiently large positive integer k, or by other means). Keeping in mind that in practical calculations only finite decimal numbers are used, we see that the conditions for applicability of the spectral method do not place any severe restriction in actual computations. Of course, it is possible to calculate with decimal numbers directly using spectra of sequences of decimal numbers as developed below.

2.6. SPECTRA OF A SEQUENCE OF FINITE DECIMAL NUMBERS. It is only necessary to explain the formation of the ordinary spectrum of a sequence of positive decimal numbers because all the generalizations derived from this notion are made for decimal numbers in the same way as for integers.

Consider the following sequence of positive decimal numbers

$$12.623 \quad 0.17 \quad 25.1 \qquad (2.15)$$

The corresponding spectral numbers for integers are numbers having the same number of digits. For decimal numbers, the spectral numbers must

have the same number of digits for the decimal part, but these two numbers may differ. Thus spectral decimal numbers of the sequence (2.15) are

12.623 00.623 25.100

or

012.623 000.170 025.100

In the first case the rhythms for the decimal and entire parts are different; in the second case, they are equal. Thus there occur two numbers having connection with the rhythm. The sum of these two numbers is called the main rhythm of the spectrum of decimal numbers and is denoted by h. The rhythm for the decimal parts is denoted by p and is called the point rhythm of the spectrum.

The ordinary spectrum of a sequence of decimal numbers is obtained by writing one after another all the spectral numbers. Thus the spectrum S for the sequence (2.15) with h = 6, p = 3 is the number

S = 012623000170025100.

The spectrum of a sequence of decimal numbers, like the spectrum of a sequence of integers, is an integer. The only difference occurs in the partitioning of such spectra. The ordinary integer spectrum with single rhythm is partitioned into sections only, and each section contains one integer. The integer spectrum with double rhythm (H and h) is partitioned first into sections (by the rhythm H) and then each section is partitioned into subsections (by the rhythm h). Every section contains one integer only. The same is true for every subsection. In the case of a spectrum of decimal numbers, the spectrum is partitioned by the main rhythm h, but every section of the spectrum of the sequence of decimal numbers contains a decimal number. Thus the main rhythm h in decimal spectra plays the same role as the rhythm h in integer spectra. The point rhythm p does not serve in further partitioning of the section into subsections. Its only purpose is to place the decimal point in each section after p digits, coutning from right to left. The spectrum S partitioned into sections and written with the decimal points is

S = 012.623|000.170|025.100

The decimal points, the signs separating the sections as well as the zero(s) at the beginning of the first section may be omitted without introducing any ambiguity. From S = 12623000170025100 it is very easy to obtain by means of h and p the sequence (2.15).

The notion of the spectrum of a sequence of positive and negative decimal numbers is obtained in a similar way as for integers because the symbols S^+ and S^- have the same meaning also in this case. This remark also applies to inverse spectrum and corrected spectra. All the specific operations with spectra mentioned above are applicable to the spectra of decimal numbers in the same way as they are for integers.

The formulae (2.6) and (2.7), are modified for polynomials with finite decimal numbers as follows:

$$S_d = 10^p\, P(10^h), \quad S_i = 10^{nh+p}\, P(10^{-h})$$

The formulae (2.8) and (2.9) remain valid. Thus, to avoid any misunderstanding, it must be always stressed whether the spectrum is the spectrum of integers, (I-spectrum) or the spectrum of decimal numbers (D-spectrum).

The partitioning of all noninteger spectra (marked with *) in the case of integers starts always in both directions (right and left) from the decimal point of the noninteger spectrum. The partitioning of a noninteger spectrum from decimal numbers starts in both directions also, but not from the decimal point of the noninteger spectrum, instead it starts from the point which is p digits to the right of the decimal point. Thus the internal operations on a D-spectrum are very similar to the internal operations on an I-spectrum described in Section 2.3 and it is not necessary to add any further explanation. The arithmetical operations with spectra are binary operations. We consider here addition and multiplication of spectra. We note that an operation of a spectrum and a number does not require special consideration because any number is considered, in the context of numerical spectral analysis, to be an I-spectrum with one section only if this number is an integer, or a D-spectrum with one section if this number is a decimal one. We also observe the general rule that the rhythm h for all the spectra used in one problem must be the same. This rhythm remains the same under any arithmetical operation.

Addition is defined only on I-spectra or on D-spectra with the same h and p. If it is necessary to add an I-spectrum and a D-spectrum, the first must be transformed into a D-spectrum (with the same p). Multiplication is defined for any two spectra. If both are D-spectra, the point rhythm will be the sum $p_1 + p_2$ of the point rhythms of the factor spectra. Thus an integer spectrum can be considered as a D-spectrum with p = 0.

3. APPLICATIONS OF NUMERICAL SPECTRA TO CALCULATIONS WITH POLYNOMIALS.

3.1. THE NUMERICAL SPECTRAL METHOD IN ALGEBRA AND ANALYSIS.

The spectral method to be developed in this paper is essentially a method of arithmetization of the algebraic or analytic problems. The analysis of such problems often reduces to a set of operators Φ_i and to the computation of elements F_i defined by

$$F_i = \Phi_i(f_1, f_2, \ldots, f_n) \qquad i = 1,2,\ldots,m \tag{3.1}$$

where each f_i is an element belonging to a set X_i. This may also be written as a single operator Φ on the space $X = X_1 \times \ldots \times X_n$ into the space $Y = Y_1 \times \ldots \times Y_m$, where $F_i \in Y_i$:

$$F = \Phi(f), \quad f = (f_1,\ldots,f_n), \quad \Phi = (\Phi_1,\ldots,\Phi_m), \quad F = (F_1,\ldots,F_m)$$

Thus the operator Φ represents the totality of operations (of any kind) to be performed on the elements f_i. The elements $F_1,\ldots,F_m$ may be interpreted as the output of a multiinput system. In the present exposition the elements f_i can be functions of one or several variables, sequences, vectors, matrices, etc.

The essence of the spectral method is to obtain the elements F_i using arithmetical operations. This end is achieved in three steps. The first step is the transformation of all the elements f_i into numbers - their spectra S_i, $i = 1,2,\ldots,n$. The second step is the calculation of a number called the resulting spectrum S of the required F. This calculation is performed by means of a formula

$$S = \Phi^*(S_1, S_2,\ldots,S_n) \tag{3.2}$$

where Φ^* is the totality of arithmetical operations to be performed. Formula (3.2) entails an arithmetization of the operator Φ_i, $i = 1,\ldots,m$. The third and last step is to find the inverse transform of S. This means the translation of the number S into the element F.

3.2. CALCULATIONS WITH POLYNOMIALS. We first consider the application of the spectral method to the computation of a linear combination of a given set of polynomials. We shall assume that the coefficients of the polynomials and the coefficients of the linear combination are integer. The problem then is to find

$$P(x) = \sum_{j=1}^{n} a_j P_j(x) \tag{3.3}$$

where $P_j(x)$, $j = 1,\ldots,n$ are given polynomials. Putting $x = 10^h$, we obtain

$$P(10^h) = \sum_{j=1}^{n} a_j P_j(10^h)$$

Thus if the rhythm h is compatible not only with the polynomials P_j but also with the resulting polynomial P, then

$$S = \sum_{j=1}^{n} a_j S_j \qquad (3.4)$$

Note that (3.3) and (3.4) are specific realizations of the abstract relations (3.1) and (3.2) respectively. The formula (3.4) suggests to establish the following rule: All the spectra of elements f_i entering in the transformation (3.1) must be made by means of the same rhythm h.

Thus the main problem is to establish the formula for calculating the resulting spectrum (in our case (3.4)) and to find the rhythm h, which will be surely compatible with P_i and P.

The determination of a rhythm compatible with the given polynomials P_i is trivial. There are two ways of finding the rhythm h compatible with the polynomial P. One is the precise and the other is a rough way. Both are based on the use of majorants.

We first introduce the notion of the indicatory number. For any positive integer p, let (p) denote the number of digits in p. The indicatory number (associated with fixed majorants) is a number δ such that (δ) is the compatible rhythm.

Let A be the maximum of the absolute values of the coefficients of the polynomials P_i, $i = 1,\ldots,n$ and $a = \max |a_i|$. Then it follows readily that

$$\delta = 2\, n\, a\, A \qquad (3.5)$$

is the indicatory number. A less precise way of obtaining a compatible rhythm is given by

$$h = (a) + (A) + [\log 2n] + 1 \qquad (3.6)$$

where [u] denotes the greatest integer in u. Note that the formula (3.6) takes into consideration the number of digits in each of the coefficients but not the values of these coefficients. We call this the rough rhythm and (δ) the precise rhythm, since in general (δ) $\leq$ h where h is rough rhythm (given by (3.6)). Each of these two methods has its advantages. The first gives a smaller rhythm but requires more calculations. The second requires only a trivial calculation but gives in general a less precise rhythm.

Example. Add the following polynomials:

$P_1(x) = 32x^2 - 18x + 32$, $P_2(x) = 54x^2 - 42x - 28$, $P_3(x) = -8x^2 + 9x + 47$.

The above mentioned numbers are

$$a = 1, \quad (a) = 1, \quad A = 54, \quad (A) = 2, \quad n = 2, \quad \delta = 324.$$

The rhythm determined in the precise way is h = 3 and in rough way h = 4. The decreasing spectra of the polynomials (with h = 3) are

$$S_1 = 31|982|032, \quad S_2 = 53|957|972, \quad S_3 = -7|990|953.$$

The resulting spectrum S is $77|949|051$, from which we get the sum of the polynomials,

$$P(x) = 78x^2 - 51x + 51.$$

It is obvious that the advantage of spectral calculations occurs only when calculating machines (desk calculators or computers) are used. The advantage increases with the capacity of the machine. The spectral method is particularly useful when many different linear combinations of the same set of polynomials must be calculated, because the spectra are to be formed only once and then retained in the memory. If the spectra are too long to enter into the machine, they can be cut and calculated part by part.

Now we consider the application of the spectral method to the calculation of the product of two polynomials:

$$P_1(x) = \sum_{i=1}^{n_1} a_i x^i, \quad P_2(x) = \sum_{i=1}^{n_2} b_i x^i$$

As before, the main problem is to determine the rhythm of the spectra. Let $A_1 = \max|a_i|$, $A_2 = \max|b_i|$, $n = 1 + \max(n_1, n_2)$. Then it is easy to show that $\delta = 2n A_1 A_2$ is the indicatory number (and therefore (δ) is the precise rhythm) and that

$$h = (A_1) = (A_2) + [\log 2n] + 1$$

is a rough rhythm.

Example. Multiply the polynomials:

$$P_1(x) = 46x^2 - 54x + 48, \quad P_2(x) = 26x^2 + 21x - 23.$$

For this example $\delta = 8424$, which leads to the rhythm $h = 4$. In the rough way, we get the wider rhythm $h = 5$. Taking $h = 4$, we obtain

$$S_1 = 45|9946|0048 \qquad S_2 = 26|0020|9977$$

$$S = S_1 S_2 = 1195|9561|9056|2249|8896$$

This spectrum leads to the desired product:

$$P(x) = 1196x^4 - 438x^3 - 944x^2 + 2250x - 1104.$$

The advantage of using spectra in this example is obvious. Instead of 9 multiplications and 4 additions and writing of intermediary results, we have only one multiplication of spectra and the determination of the rhythm. The spectra can be formed directly by typing the digits into the calculating machine. Note that there is an additional advantage in the spectral method in that the rhythm is calculated only once and may be used for a large number of calculations of the same kind. There is no need to copy down the resulting spectrum from the machine since the resulting polynomial can be written directly; so spectra occur only in the calculating machine.

3.3. ADVANTAGES OF SPECTRAL CALCULATIONS. The last example gives us the possibility to formulate some advantages of the spectral method which are more or less valid in all spectral calculations.

For desk calculators these advantages are:

1. The number of operations is considerably reduced.
2. The writing of intermediary results is reduced or eliminated completely.
3. The scheme of calculating is much simpler.
4. The number of possible mistakes is reduced.

For computers the advantages are:

1. The number of operations is much less.
2. The program is simpler.
3. The required memory capacity is considerably less.

4. APPLICATIONS OF NUMERICAL SPECTRA TO ARITHMETIC.

4.1. REARITHMETIZATION OF ARITHMETIC.

The application of spectra to algebra was described earlier as an arithmetization of the algebraic problems. So the applicaton of spectra to arithmetic seems to be, at the first glance, meaningless. To arithmetize arithmetic may appear to be of no use. But this is not the case. To explain the usefulness of the application of numerical spectra to arithmetic, we consider the problem of finding the following sums

$$\sum_{j=1}^{n} a_j, \qquad \sum_{j=1}^{n} b_j, \qquad \sum_{j=1}^{n} c_j$$

Each sum is obtained by n-1 additions; thus the total number of arithmetical operations is 3(n-1). The numbers a_j, b_j, c_j of the same index can be consiaered as the coefficients of the polynomial

$$P_j(x) = a_j + b_j x + c_j x^2, \quad j = 1,2,\ldots,n.$$

By adding these polynomials, we obtain

$$P(x) = \sum_{j=1}^{n} P_j(x) = \sum_{j=1}^{n} a_j + x \sum_{j=1}^{n} b_j + x^2 \sum_{j=1}^{n} c_j.$$

The coefficients of the resulting polynomial are just the required sums. Thus the required sums are obtained by means of n-1 algebraic operations. This apparent decrease in the number of operations is surely only formal, because in ordinary calculations we must perform the same number 3(n-1) or arithmetical operations. But by means of numerical spectra every polynomial $P_j(x)$ is replaced by its spectrum S_j and we have to perform n-1 additions of spectra. This means that we have to perform only n-1 arithmetical operations to obtain the resulting spectrum $S = S_1 + \ldots + S_n$. It is evident that the spectrum S is just the spectrum of the polynomial P(x). The values of the sections of the spectrum S are the values of the required sums.

Thus this rearithmetization of arithmetic happens to be useful in reducing considerably the number of operations. The use of algebra is only necessary for the justification of several spectral processes. The spectrum S_j of numbers a_j, b_j, c_j with the same index, can be obtained directly without recourse to any algebra.

We now turn to systematic formulation of the spectral method in arithmetic.

4.2. ADDITION AND SUBTRACTION. The problem of addition was elaborated in Section 4.1 as an example of the advantages of the spectral method in arithmetic. It remains only to find a compatible rhythm for this operation. Let a_{ij}, $1 = 1,\ldots,m$, $j = 1,\ldots,n$ be a given set of positive numbers. We want to find the sums

$$K_j = \sum_{i=1}^{m} a_{ij}, \quad j = 1,\ldots,n.$$

The indicatory number is $\Delta = ma$, where $a = \max a_{ij}$. The rhythm h obtained in the rough way is $h = (a) + [\log m] + 1$. For each $j = 1,\ldots,n$, let S_j denote the spectrum of the sequence,

$$a_{1j}, a_{2j},\ldots,a_{mj}$$

Then the values of the sections of the spectrum $S = \sum_{j=1}^{n} S_j$ are the required sums K_j, $j = 1,\ldots,n$.

In the case of subtraction the number n remains arbitrary and $m = 2$. The indicatory number is $\Delta = a$ and the rough rhythm is $h = (a)$. Thus the subtraction in spectral arithmetic is a simpler operation than addition. It may be noted that the advantage of using spectra for addition and subtraction is not as great as it is in the case of multiplication or in problems involving both addition and multiplication.

Example: (a). Perform the following additions:

1628	2834	1986	1153	2020
839	1723	1312	992	813
1213	752	678	523	1412

Here $a = 2834$, $n = 3$, $\Delta = 8502$, $h = 4$.

The same rhythm is obtained in the rough way.

$$S_1 = 1628|2834|1986|1153|2020$$

$$S_2 = 839|1723|1312|0992|0813$$

$$S_3 = 1213|0752|0678|0523|1412$$

$$S = 3680|5309|3976|2668|4245$$

(b). Perform the following subtractions:

383 - 168, 96 - 57, 215 - 182, 312 - 89

a = 383, (a) = 3, h = 3.

S_1 = 383|096|215|312

S_2 = 168|057|182|089

$S = S_1 - S_2$ = 215|039|033|223.

The sections give the desired answers.

4.3. SCALAR MULTIPLICATION AND DISTRIBUTIVE MULTIPLICATION.

We consider the operation of multiplying each number of a given sequence $a_1,\ldots,a_n$ of positive numbers by a fixed positive number b.

It is easy to see that the indicatory number is δ = ab, where $a = \max a_j$. The rhythm in the rough way is h = (a) + (b). The values of the sections of the resulting spectrum Sb give the desired sequence $ba_1,\ldots,ba_n$.

Example: Multiply each of the numbers 68, 105, 87, by 93.

Here a = 105, δ = 9765, h = 4. In the rough way the rhythm is 5. Using the precise rhythm we get

S = 68|0105|0087

b = 93

S.b = 6324|9765|8091

We now consider the operation of multiplication of each number of a sequence with each number of another sequence (distributive multiplication). Let a_i, i = 1,...,n and b_j, j = 1,...,k be given sequences of positive numbers. We elongate the first sequence by putting (k-1) zeros between each two terms. We then form the ordinary spectrum S_1 of the elongated sequence and the ordinary spectrum S_2 of the second sequence. Then $S = S_1\ S_2$ gives the spectrum of the desired sequence of products. Note that the indicatory number is δ = ab, where $a = \max a_i$, $b = \max b_j$, and the rhythm in rough way is h = (a) + (b).

Example. Multiply each number of the sequence 18, 16, 11 by the numbers 47, 33, 28.

In this example, n = k = 3, a = 18, b = 47, δ = 846, h = 3 (in rough way it would be h = 4).

$$S_1 = 18|000|000|016|000|000|011$$

$$S_2 = 47|033|028$$

$$S = S_1 S_2 = 846|594|504|752|528|448|517|363|308$$

Hence the desired products are given by sections in the same order from left to right, for example 18 x 47 = 846, 18 x 33 = 594, 18 x 28 = 504, etc.

4.4. CALCULATIONS WITH FRACTIONS. The sum of positive fractions

$$\frac{a_1}{b_1} + \frac{a_2}{b_2} + \ldots + \frac{a_n}{b_n} \qquad (4.1)$$

may be obtained by the spectral method as follows.

We form the ordinary spectra of these fractions, $S_1, S_2, \ldots, S_n$ with the rhythm

$$h = [\log n A^n] + 1 \qquad (4.2)$$

where $A = \max (a_i, b_i)$, $i = 1, \ldots n$. Then the last section of the product spectrum $S = S_1 S_2 \ldots S_n$ is the denominator and the penultimate section is the numerator of the sum of fractions. The calculation of the product spectrum may be simplified by retaining after each multiplication the last two sections only.

If some of the terms of the expression (4.1) are negative, then we associate the negative sign with the denominators of these terms; in such cases of mixed addition and subtraction the rhythm is slightly changed:

$$h = [\log 2n A^n] + 1, \quad A = \max (|a_i|, |b_i|) \qquad (4.3)$$

The application of spectra always reduces the number of operations. Ordinarily the sum (4.1) is calculated by means of n divisions and n - 1 additions. Using the spectral method the result, in the form of a

decimal number, is obtained by means of n multipl. [illegible] and one division only. For computers the time for the perf[illegible] of division is much greater than the time for multiplicat[illegible], the saving of time will be considerable.

Example:

$$\frac{19}{31} + \frac{27}{28} + \frac{21}{17}$$

Here h = 5, S_1 = 19 00031, S_2 = 27|00028

S_1S_2 = ...|001369|00868, S_3 = 21 00[illegible]

$S_1S_2S_3$ = ...41501|14756

Thus the sum is $\frac{41501}{14756} \approx 2.8308$

Note that as a by-product we obtain the sum of t[illegible] fractions.

Inspite of the great saving of time, the met[illegible] to be not very helpful, because calculation with ordinary fractions is very seldom. But every division of decimal number [illegible] recorded as an ordinary fraction. Also it is possible to form[illegible] from decimal numbers directly. Thus $\frac{a}{b}$ represents the quo[illegible] decimal numbers and the expression (4.1) is a combination of [illegible] and additions just as the dot product is a combination of m[illegible]ion and additions.

It is easy to see that the main rhythm for [illegible]-spectrum is

$$h = [\log 2n\, A^n] + 1$$

where $A = \max(10^{r_i}|a_i|, 10^{s_i}|b_i|)$, r_i(resp. [illegible] of digits in a_i(resp. b_i). The point rhythm is

$$p = [\log B] + 1$$

where $B = \max(10^{r_i}\{a_i\}, 10^{s_i}\{b_i\}$ and $u\{[illegible]$ decimal part of u.

Example:

$$\frac{12.19}{8.12} + \frac{6.14}{10.25}$$

For this case, n = 2, h = 8, p = 2.

$$S_1 = 12.19\ 000008.12$$

$$S_2 = 6.14\ 000010.25$$

$$S_1S_2 = \ldots|0174.8043|0083.2300$$

Thus the result is $\frac{174.8043}{83.2300} \approx 2.1003$

4.5. INTERVAL ARITHMETIC AND NUMERICAL SPECTRA

For given real numbers a,b with $a \leq b$, the set $\{x: a \leq x \leq b\}$ is called an interval number and is denoted by [a:b]. Note that the degenerate case a = b is included. Interval arithmetic is an arithmetic system which uses interval numbers as elements. Interval arithmetic operations are defined by

$$[a:b] \circ [c,d] = \{z: z = xoy,\ a \leq x \leq b,\ c \leq y \leq d\}$$

where o denotes addition, subtraction, multiplication or division (division by an interval containing zero is excluded). Since the ordinary arithmetic operations are continuous, they map the cartesian product [a:b] x [c:d], which is a compact connected set, onto a compact connected set, i.e., a closed real interval. In fact, the following formulae can be easily deduced:

$$\begin{aligned} [a:b] + [c:d] &= [a+c : b+d] \\ [a:b] - [c:d] &= [a-d : b-c] \\ [a:b] \cdot [c:d] &= [\min(ac, ad, bc, bd) : \max(ac, ad, bc, bd)] \\ [a:b] \div [c:d] &= [a:b] \cdot [\tfrac{1}{d}:\tfrac{1}{c}] \quad \text{if } 0 \notin [c:d] \end{aligned} \tag{4.4}$$

Thus for interval multiplication, four ordinary multiplications are to be performed in the case $a < 0 < b$, $c < 0 < d$, whereas two multiplications suffice for the remaining cases.

It follows from these formulae that under the correspondence $[a:a] \leftrightarrow a$, the ordinary arithmetic of real numbers is included in interval arithmetic if one makes the identification a = [a:a]. For properties of interval arithmetic, see, for instance, [11], [12], [22].

It should be noted that numbers that arise in applications are actually interval numbers due to error in experimental initial data or round off in performing arithmetic operations. It is not surprising

therefore that interval arithmetic should find its way, as it has recently [6], to elementary calculus books.

Interval arithmetic plays an important role in the automatic analysis and control of error in digital computations [11] and provides a convenient setting for the accurate numerical integration of ordinary differential equations [12]. It may also be used advantageously in the programming of Newton's method for automatic computation [22]. An extensive bibliography on the theory and applications of interval arithmetic and interval functions is given in [11], [12]. In this section we shall show that numerical spectra can be used to an advantage in interval arithmetic.

It follows from above that addition of n intervals $[a_i:b_i]$ reduces to finding certain linear combinations of numbers, which was elaborated earlier. Thus the indicatory number is $A = 2\,n\,\alpha$, where $\alpha = \max\{|a_i|, |b_i|\}$ and the rhythm in the rough way is $h = (\alpha) + [\log 2n] + 1$. The spectral process is to form the ordinary spectrum S of the sequence $a_1, b_1, \ldots, a_n, b_n$ with rhythm h and to multiply S with a number consisting of n digits of 1 interlaced with n-1 digits of 0 between every two 1's. This gives a spectrum with an even number of sections; the effective values of the middle section M_1 and M_2 are the beginning and the end of the sum of intervals.

Example: [8:16] + [-5:8] + [-16:-3]. Here h = 2.

```
        S = 8|15|95|07|83|97
            1|0001|0001
  8|16|03|23|87|20|79|04|83|97
```

The effective values of the middle sections are -13, 21, which give the required interval sum. Note that the resulting spectrum also gives the sum of the first two (last two) intervals which may be read from the second and fourth (resp. seventh and eiththh) sections of the resulting spectrum. The addition of four intervals using spectra gives five results and so on.

Subtraction of intervals can be transformed into addition:

$$[a:b - [c:d] = [a:b] + -d: -c]$$

The advantages of spectral addition and subtraction of intervals are limited to desk calculators. Spectra are particularly useful when all, or most of, the obtained interval sums are needed. This advantage

can be improved by the appropriate arrangements of the intervals, since the commutative and associative laws are valid for interval addition.

In view of the formula for interval product given in (4.4), the spectral process for obtaining such a product was elaborated earlier (Section 4.3). This product may also be obtained by use of inverse spectra, which would avoid the negativity of spectra. In this case we must elongate the first sequence in the middle by one zero. The interval product is then given by the maximum and minimum values of $\Sigma_1 \Sigma_2$, where Σ_1 and Σ_2 are the inverse spectra of the first and second intervals.

4.6. CHECK UP OF THE RESULTS. We shall confine the check up by spectra to the traditional tests by 9, and by 11. The check up by 9 consists of finding the sums $s_i = s_i(a_i)$ of the digits of each number a_i and of performing all the required operations with t_i, where $t_i \equiv s_i \pmod 9$, instead of a_i. To perform this check up by means of spectra, we have to use spectra to calculate each of the numbers s_i. This may be obtained as a dot product, where one of the n-dimensional vectors has all its coordinates equal to one. We use the rhythm $h = 2$ and multiply the spectrum of a number a_i (having n digits) by the spectrum of the sequence 1, 0, 1, 0, ..., 1 of $2n - 1$ terms. The middle part M(S) of the resulting spectrum gives the corresponding number s_i.

The check up by 11 is performed in the same way except that instead of using the spectrum of the sequence 1, 0, 1, 0, ..., 1, we use the spectrum of the sequence 1, -1, 1, -1, ..., 1 of $2n - 1$ terms.

5. APPLICATIONS OF NUMERICAL SPECTRA TO COMPUTATIONAL METHODS OF LINEAR ALGEBRA.

5.1. COMPUTATION OF INNER PRODUCT USING SPECTRA.

Let a and b be two vectors in R_n with components a_j and b_j, $j = 1, \ldots, n$. We introduce the auxiliary polynomials

$$P(x) = \sum_{j=1}^{n} a_j x^{n-j}, \quad Q(x) = \sum_{j=1}^{n} b_j x^{j-1}$$

and

$$R(x) = P(x)Q(x) = \sum_{j=1}^{2n-1} c_j x^{2n-j-1}$$

It is easy to verify that $c_n = \sum_{j=1}^{n} a_j b_j = a.b$.

Now we choose the rhythm h to be compatible with the polynomial R(x). This will occur if $10^h = 2n\alpha\beta$ is an indicatory number, that is, if

$$2n\alpha\beta < 10^h \tag{5.1}$$

where $\alpha = \max |a_j|$, $\beta = \max |b_j|$. Since

$$R(10^h) = P(10^h)\, Q(10^h) \tag{5.2}$$

and the rhythm h is compatible also with the polynomials P(x) and Q(x), it follows that (5.2) is the relation between the decreasing spectra S, S_1, S_2 of the polynomials R(x), P(x), Q(x), respectively, that is, $S = S_1 S_2$. Thus the middle coefficient c_n of the polynomial R(x) is the effective value of the middle section of the spectrum S.

$$c_n = a.b \quad M(\quad) = M(S_1 S_2).$$

On the other hand S_1 is the ordinary spectrum of the vector a and is to be denoted by S_a, S_2 is the inverse spectrum of the vector b and is to be denoted by S_b. Thus

$$a.b = M(S_a \quad S_b)$$

is the formula for the computation of the standard inner product in R_n using the spectral method.

5.2. MATRIX ALGEBRA USING SPECTRA. Let A and B be two m x n matrices To compute A + B by the spectral method we need to form the spectra of the matrices A and B. It is necessary that both spectra be of the same kind (that is, both by rows or both by columns) and be obtained by means of the same rhythm h. If h_A is a rhythm compatible with the matrix A and h_B with the matrix B, then

$$h = \max (h_A, h_B) + 1$$

is compatible rhythm with A + B.

Another value of a compatible [illegible] from the values of the coefficients a_{ij} and b_{ij}. [illegible] $= \max |b_{ij}|$ and $\gamma = \max (\quad, \quad)$. Then the index [illegible] + 2. and the rough method of determining h [illegible] m x n matrix and B be an n x p matrix. Then are [illegible] to obtain the product C = AB using spectra.

1. Vectorial Method. The element c_{ij} in the matrix C is obtained as the dot product of the ith row vector of A with the jth column vector of B. Thus we have to form the inverse spectrum of the ith row vector and to multiply it consequently with the ordinary spectra of the vectors of the first, second, ..., nth column. The middle part of such products will give us all the elements of the ith row of the resulting matrix. The indicatory number is $\delta = 4$ [illegible] and the rough rhythm is $h = (\alpha) + (\beta) + 1$, where α and β are the same as defined above.

2. First Matrix Method. After determining the rhythm h, we take $H = (2n - 1)\,h$ and form the ordinary double rhythm row spectrum of the first matrix and multiply it with the inverse spectrum of the jth column of B, $j = 1,\ldots,n$. This product is partitioned into sections by the rhythm H and then into subsections by the rhythm h. Thus every section, being a spectrum, has its middle part. The effective values of the middle parts of all the sections are the elements of the jth column of the product AB.

3. Second Matrix Method. By this method, all the elements of the product AB are obtained by one multiplication. For this method we need three rhythms: h and H as defined previously, and $H' = pH$. We form the double rhythm spectrum S_A of the matrix A with rhythms H' and h. We also form the inverse spectra Σ_j of the column of B with rhythm h. Finally we form the ordinary spectrum S_B of this sequence of numbers $\Sigma_1, \Sigma_2, \ldots, \Sigma_n$ with rhythm H. We partition $S = S_A S_B$ into sections by the rhythm H' and then partition each section into subsections by the rhythm H and finally partition each subsection into parts by the rhythm h. Each subsection, being itself a spectrum, has a middle part. The effective values of these middle parts give the elements of AB arranged row by row from left to right.

Example: Find AB using the first matrix method.

$$A = \begin{pmatrix} 1 & 0 & 2 \\ 1 & 3 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 1 & 2 \\ 0 & 1 \\ 3 & 1 \end{pmatrix}$$

The indicatory number is obtained using (5.1). [illegible] = 54, h = 2 and H = 10. The ordinary spectrum of the matrix A is S = 1|00|02|00|00|01|03|00. The inverse spectra of the first and second column vectors are Σ_1 = 3|00|01, Σ_2 = 1|01|02.

$S\Sigma_1$ = 3|00|07|00|02|03|09|01|03|00

$S\Sigma_2$ = 1|01|04|02|04|01|04|05|06|00

The resulting matrix is $\begin{pmatrix} 7 & 4 \\ 1 & 5 \end{pmatrix}$.

5.3. EVALUATION OF DETERMINANTS AND SOLUTIONS [illegible] OF LINEAR ALGEBRAIC EQUATIONS USING SPECTRA.

We shall apply the spectral method to calcul[illegible] determinant D of order n, whose coefficients are integers or [illegible] numbers. For purposes of the spectral method, it is advant[illegible] ber D as the determinant of the n x n matrix $A = [a_{ij}]$. [illegible] te with the matrix A a column matrix C_n whose ith row [illegible] ordinary spectrum S_i of the ith row of ^nA.

Applying the "pivot" method to the elemen[illegible] $a_{1n} \neq 0$; otherwise rearrange rows), we obtain

$$\text{Det } A = \frac{(-1)^{n+1}}{a_{1n}^{n-2}} \text{ Det } A'$$

where the elements of the matrix A' are given by

$$a'_{jk} = a_{1n}a_{jk} - a_{1k}a_{jn}, \quad j = 1,2,\ldots,n-1, \quad [illegible] \quad (5.3)$$

We also associate with the matrix A' a column [illegible], whose row element is $\tilde{S}'_i$ when ~ denotes the ablation [illegible] section of the ordinary spectrum S'_i of the ith row of A'. [illegible] it follows readily that

$$S'_{i-1} = S_i a_{1n} - s_1 a_{in} \quad i = 2,\ldots, \quad (5.4)$$

Thus to the reduction of the determinant of [illegible] a determinant of order n - 1 (i.e., from det A to det A'), [illegible] trans-formation of the column matrix $(S_1,\ldots,S_n)^t$ [illegible] $\frac{(-1)^{n+1}}{a^{n-2}}$ $(\tilde{S}_1,\ldots,\tilde{S}_n)^t$ where [illegible] is the effecti[illegible] section of the spectrum S_1. Continuing inductively in [illegible] obtain a column matrix composed of one number. It is [illegible] is number is equal to det A.

It should be noted that the ordinary "p[illegible] the reduction of a determinant of order n t[illegible] order n - 1, a total of $2n^2 - 3n + 1$ operations, whereas [illegible] requires only 5n - 4 operations, the formation of each [illegible] s one operation.

Example. Evaluate using spectra:

$$D = \begin{vmatrix} 2 & 3 & 1 & 2 \\ 4 & 2 & -1 & 3 \\ 5 & -2 & -3 & 1 \\ 2 & 1 & 3 & 2 \end{vmatrix}$$

The indicatory number for forming spectra is [illegible], where $a = \max|a_{ij}|$. Note that the rhythm (δ) is compatible with all the spectra occuring in the calculations of the above recursive reductions. Using the rhythm h = 4, we obtain

$$C_4 = \begin{bmatrix} 2|0003|0001|0002 \\ 4|0001|9999|0003 \\ 4|9997|9997|0001 \\ 2|0001|0003|0002 \end{bmatrix}$$

The first transformation leads to the following [illegible]:

$$-\frac{1}{4} \cdot \begin{bmatrix} 1|9994|9995 \\ 7|9992|9993 \\ 0|0003|9996 \end{bmatrix}$$

Continuing we obtain det A = -52.

Numerical spectra can be used advantageously [illegible] number of direct and iterative methods for solving [illegible] algebraic equations (see, for instance, [3], [4], [20], [illegible]). This distribution accrues from the fact that these methods use [illegible] operations (dot product, addition and multiplication of [illegible], linear combinations, evaluation of determinants, etc.) which were discussed in the previous sections. Thus the spectral method [illegible] instance in the Gauss-Jordan elimination procedure, the [illegible] orthonormalization procedure, etc. The determination [illegible] algorithms does not present any difficulties.

6. APPLICATION OF NUMERICAL SPECTRA TO [illegible] EQUATIONS - GRAEFFE'S METHOD

The well-known Graeffe's method ([5], [6], [illegible]; the

last reference contains an extensive bibliography) can be simplified by means of numerical spectra. The simplification can be made in the first part of Graeffe's method, that is, in transformation of the polynomial equation. Two spectral methods are given in the present paper.

6.1. FIRST SPECTRAL METHOD.

The first step in Graeffe's method is to transform the polynomial P(x) of the equation

$$P(x) = \sum_{i=0}^{n} a_i x^{n-i} = 0$$

into the polynomial $P_1(x)$ having as zeros the squares of zeros of the previous polynomial P(x). The spectra to be used are the following: the ordinary decreasing spectrum S of the polynomial P(x) which is given by

$$S = P(10^h) \tag{6.1}$$

with the compatible rhythm h, and the corrected spectra $\bar{S}$ on all the even places of the same polynomial P(x). This spectrum $\bar{S}$ is in fact the ordinary decreasing spectrum of the corrected polynomial $\bar{P}(x)$ on all the even places of P(x): $\bar{P}(x) = (-1)^n P(-x)$. Thus

$$\bar{S} = \bar{P}(10^h) = (-1)^n P(-10^h) \tag{6.2}$$

We now state the fundamental theorem in the spectral processes of the transformation of the equations.

Theorem 6.1. Let S be the ordinary decreasing spectrum of the polynomial P(x), $\bar{S}$ be the corrected decreasing spectrum of the same polynomial, both spectra being formed with rhythm h, and let S_1 be the ordinary decreasing spectrum of Graeffe's transform, with rhythm 2h. Then

$$S_1 = S\,\bar{S} \tag{6.3}$$

The rhythm h is given by the formula

$$h = [\log a\ (n+1)] + 1 \tag{6.4}$$

where $a = \max |a_i|$ in case all the coefficients a_i are integers and $a = \max |a_i\ 10^{r_i}|$ in case some of the coefficients are decimal numbers (r_i is the number of decimals of a_i). The point rhythm, in the latter case is given by $p = [\log b] + 1$, where $b = \max |a_i\ 10^{r_i}|$.

Proof: It is known that the relation between Graeffe's transform-polynomial $P_1(x)$ and the polynomial $P(x)$ is

$$P_1(x^2) = (-1)^1 P(x) P(-x).$$

Let h be a positive integer such that the rhythm h is compatible with $P(x)$ and the rhythm 2h is compatible with $P_1(x)$. Putting $x = 10^h$ in the above relation and using (6.1) and (6.2) we obtain $S_1 = S\bar{S}$. This proves (6.3).

Now we prove that the value h given by (6.4) is a compatible rhythm. For a rhythm h to be compatible with the polynomial $P(x)$, it is sufficient that h satisfies the inequality

$$h > \log a + \log 2 \tag{6.5}$$

For the rhythm 2h to be compatible with the polynomial $P_1(x)$, it is sufficient that h satisfies the inequality $10^h > 2(n+1)\, a^2$ which leads to the inequality

$$h > \log a + \frac{1}{2} \log 2(n+1) \tag{6.6}$$

It is obvious that h given by (6.4) satisfies (6.5) and (6.6). Thus the numbers h and 2h are rhythms compatible with $P(x)$ and $P_1(x)$ respectively. The proof of the validity of formula (6.4) for decimal numbers and the point rhythm requires only a minor modification of the above proof. The proof of the theorem is completed.

Remark. Graeffe's method requires the successive formation of a sequence of polynomials $P(x)$, $P_1(x)$, $P_2(x)$, ..., $P_k(x)$, where each polynomial $P_i(x)$ $(i = 1,\ldots,k)$ is the Graeffe transform of the previous one. Once the last $P_k(x)$ has its zeros separated enough for the required accuracy the transformation is finished. All the consecutive Graeffe transforms $P_1(x),\ldots,P_k(x)$ will have the following compatible rhythms $2h,\ldots,2^k h$ if h is obtained from (6.4).

We shall now explain a procedure for forming the corresponding sequence of spectra

$$S_1, S_2,\ldots,S_k \tag{6.7}$$

without the formation of any intermediary polynomial $P_i(x)$. We first form the spectra S and $\bar{S}$ directly from the given polynomial $P(x)$. The ordinary decreasing spectrum S_1 of the polynomial $P_1(x)$ in order to

obtain S_2 and then to continue the formation of the sequence (6.7). This can be done in the following way. We find the effective values of all even sections of the spectrum S_1'. Afterwards we subtract from S_1 twice these effective values on the corresponding places. This gives the spectrum $\bar{S}_1$ and we can continue the spectral process until S_k is obtained. From S_k we obtain the polynomial $P_k(x)$ and proceed then in the classical way.

Example: We illustrate the method by making two consecutive Graeffe's transformations for the equation

$$x^6 + 3x^5 - 2x^4 + 5x^3 + 4x^2 - 3x + 1 = 0$$

Here $h = 2$, $S = 1|02|98|05|03|97|01$

$\bar{S}$ can be obtained directly from $P(x)$, but we shall obtain it from S. The effective values of the even sections are 3,5,-3, so $\bar{S}$ is obtained in the following way

$$S = 1|02|98|05|03|97|01$$
$$\quad\quad -6 \quad -10 \quad +6$$

$$\bar{S} = 96|97|95|04|03|01$$

$$S_1 = S\bar{S} = 9986|9981|9979|0041|9999|0001$$

Since a spectrum can never begin with a big section, the first section in S_1 must be preceded by a section of nominal value 0. Another way to explain the necessity for this revision of S_1 is that S_1, being the spectrum of a polynomial of the sixth degree, must have seven sections.

It is obvious that the rhythm is doubled with every Graeffe's transform, which is inconvenient. But we have the operation of condensation of spectra, which can be performed if each section of the spectrum begins with p digits of 0 or p digits of 9. The rhythm in this case can be condensed by p-1 units, or possibly p units (see section 2.3). In the example above, the condensation can be made to the rhythm 2.

The spectrum $\bar{S}_1$ is obtained from S_1 in the same way as $\bar{S}$ is obtained from S. This gives

$$\bar{S}_1 = 1|12|82|42|01|01$$

$$S_2 = S_1\bar{S}_1 = 0|9794|9861|8023|1686|0083|0001$$

From S_2 we obtain the polynomial

$$P_2(x) = x^6 - 205x^5 - 138x^4 - 1997x^3 + 1686x^2 + 83x + 1$$

We remark that usually the rhythm for S_k is less than $2^k h$.

6.2. SECOND SPECTRAL METHOD. The number of operations which are needed to perform one Graeffe's transformation increase with n^2. The same number for the first spectral method increases with n. For the spectral method to be introduced in this section, the number of operations is constant, eleven operations only independent of the value of n. This fact is of theoretical interest and also provides a possibility to use simpler programs for computers.

The method is based on the operations of choice applied to the spectra. The corresponding operations of choice applied to the polynomial P(x) may be recalled from Section 2.3.

Theorem 6.2. The decreasing ordinary spectrum S_1 of the Graeffe transform $P_I(x)$ with the rhythm 2h is given by the formula

$$S_- = (S_1 + S_2)\ (S_1 - S_2) \tag{6.8}$$

where S_1 and S_2 are respectively the decreasing ordinary odd spectrum and even spectrum of the polynomial P(x) with the rhythm h. The number h is the same as in Theorem 6.1.

Proof: We start from the relation $P(x^2) = (-1)^n P(x)P(-x)$.
From $P(x) = P_1(x) + P_2(x)$ and $P(-x) = (-1)^n[P_1(x)-P_2(x)]$, we obtain

$$P_I(x^2) = [(P_1(x) + P_2(x)]\ [P_1(x)-P_2(x)]$$

Putting $x = 10^h$ we obtain

$$S_I = (S_1 + S_2)\ (S_1 - S_2).$$

The proof of the compatability of the rhythms h and 2h was given in Theorem 6.1.

The spectral process of forming the sequence of spectra

$$S_I,\ S_{II},\ S_{III},\ \ldots \tag{6.9}$$

of Graeffe's transforms is as follows. First we form by the operations of choice the spectra S_1 and S_2 from the given polynomial P(x). The spectrum S_I of the first Graeffe transform is obtained from (6.8). Then by the operation of internal or formal rounding off, we obtain the value S_I^+. Then $S_I^- = S_I^+ - S_I$. Applying four operations of choice we obtain from S_I^+ and S_I^- the following four spectra S_{I1}^+, S_{I2}^+, S_{I1}^-, S_{I2}^-. Then by two subtractions we obtain

$$S_{Ij} = S_{Ij}^+ - S_{Ij}^-, \quad j = 1,2.$$

Finally by one addition, one subtraction and one multiplication of spectra we obtain

$$S_{II} = (S_{I1} + S_{I2})(S_{I1} - S_{I2}).$$

Thus by means of the eleven operations mentioned above, we have obtained the spectrum S_{II} from the spectrum S_I. The method can be continued in the same way.

Example: We consider the same example which was solved by the first spectral method. S_1 and S_2 are obtained directly from the polynomial, or by means of S_j^+, S_j^-, $j = 1,2$ which have the following values

$S_1^+ = 1|00|00|00|04|00|01$ $\qquad$ $S_1^- = 2|00|00|00|00$

$S_2^+ = 3|00|05|00|00|00$ $\qquad$ $S_2^- = 3|00$

$S_1 = 99|98|00|04|00|01$

$S_2 = 3|00|04|99|97|00$

$S_1 + S_2 = 1|02|98|05|03|97|01$

$S_1 - S_2 = 96|97|95|04|03|01$

$S_I = 0|9986|9981|9979|0041|9999|0001$

By condensation to the rhythm 2, we obtain

$$S_I = 0|86|81|79|41|99|01$$

Applying internal round off we have

$$S_I^* = 1|00|00|00|42|00|01$$

$$S_I^- = S_I^+ - S_I = 0|13|18|21|00|01|00$$

$$S_{I1}^+ = 1|00|00|00|42|00|01$$

$$S_{I2}^+ = 0$$

$$S_{I1}^- = 18|00|00|00|00$$

$$S_{I2}^- = 13|00|21|00|01|00$$

$$S_{I1} = 0|99|82|00|42|00|01$$

$$S_{I2} = -13|00|21|00|01|00$$

Finally we get

$$S_{II} = 0|9794|9861|8023|1686|0083|0001$$

TABLE 1

Comparison of the Number of Operations

Operation	Ordinary Method	First Spectral Method	Second Spectral Method
Multiplication of coefficients	$[\frac{(n+2)^2}{4}]$	None	None
Spectra	None	1	1
Multiplication by the number 2	$[\frac{n^2}{4}]$	$[\frac{n+1}{2}]$	0
Addition (or Subtraction)	$[\frac{n^2}{4}]$	$[\frac{n+7}{2}]$	5
Other operations	None	None	5
Total	$[\frac{1}{4}(3n^2+4n+4)]$	n+4, n even n+5, n odd	11

The table shows the great difference in the number of required operations between the classical method and the spectral methods. The difference between the first and second spectral methods is not great. For $n \leq 6$ the first spectral method has a slight advantage over the second. However, even for small n, the simplicity of the program for computers puts the second method on the first preference.

7. USE OF SPECTRA IN RECURSIVE CALCULATIONS. We have dealt so far with fixed problems, that is with problems whose data were known from the outset. In contrast, calculations using recurrence relations involve results that enter as new data at each stage of such calculations. For the spectral method this means that the starting spectra will not be static. As an example of the application of numerical spectra to computations involving recurrence relations we consider difference equations.

7.1. APPLICATIONS OF NUMERICAL SPECTRA TO DIFFERENCE EQUATIONS.

Consider the second order difference equation

$$y_i = ay_{i-1}^2 + by_{i-1}y_{i-2} + cy_{i-2}^2 + dy_{i-1} + ey_{i-2} + nhx_i$$

The main step in the application of the spectral method to the above equation is to form, from the quantities entering in the equations, two sequences so that the middle part of the product of their spectra will correspond to the right side of the above equation. An example of such sequences is

(I) $y_{i-1}, y_{i-1}, y_{i-2}, d, e, nhx_i, 0$

(II) $0, \quad 1, \quad y_{i-2}, y_{i-1}, cy_{i-2}, by_{i-2}, ay_{i-1}$

It is obvious that this pair of sequences is not unique. After finding the appropriate rhythm, we form the ordinary spectra S_I and S_{II} of the sequences (I) and (II). Then it follows that

$$y_i = M(S_I S_{II})$$

The advantage of the spectral calculations is again evident: the number of operations is considerably less than in the classical method. In this case instead of 10 multiplications and 5 additions, there is just one multiplication of spectra, 4 multiplications with constants (a,b,c,nh) and obviously the extraction of the middle part.

If the spectra are too long, it is possible to divide the problem into two or more parts. For example we can multiply the spectra of the following two pairs of sequences

(I') $y_{i-1}, y_{i-1}, y_{i-2}$

(II') $cy_{i-1}, by_{i-1}, ay_{i-1}$

and

(I") d, e, n h x_i

(II") 1, y_{i-2}, y_{i-1}

We then add the obtained spectra and extract the middle part.

For desk calculations it is better to arrange the sequences so that one of them will be composed of constant quantities only. Thus for (I") and (II") it is better to arrange in the form

d, e, 1

n h x_i, y_{i-2}, y_{i-1}.

To demonstrate the application of spectra of decimal numbers we shall solve the following linear nonhomogeneous difference equation of the second order

$$y_i = 0.5y_{i-1} + 0.5y_{i-2} - 0.4(2i-1)$$

subject to the initial conditions $y_o = -0.2$, $y_i = 0.3$. The above difference equation can be represented by the following two sequences

0.5	0.5	-0.4
0.1(2i-1)	y_{i-2}	y_{i-1}

The numbers in this problem are decimal numbers. Thus we shall form D-spectra. In the case of integers, the rhythm would be $h = [\log 2n\,ab] + 1$ where n is the number of terms in any sequence, and $a = \max|a_i|$, $b = \max|b_i|$ (a_i and b_i are the terms of the first and second sequences respectively).

One of the purposes of this example is to show how to obtain the main rhythm h and the point rhythm p for the D-spectrum. Let p_i denote the number of decimals in a_i and q_i the number of decimals in b_i. Then the point rhythm p is given by $p = \max(p_i, q_i)$. The main rhythm is obtained from the same formula, where the quantities a and b are replaced by

$$a = \max\{|a_i|\}\,10^{p_i}, \quad b = \max\{|b_i|\}\,10^{q_i}$$

The spectrum of the first sequence is $S_1 = .5|0.4|9\ 6$, the second spectrum is $S_2 = .2|9.8|0.3$ and the product of spectra is $S_1S_2 = 1504932288$. The point rhythms are to be added since both spectra are D-spectra. Thus the point spectrum of the product is p = 2. The main rhythm remains the same. Thus the partitioning of the resulting spectrum gives

$$.15|.04|.93|.22|.88$$

and

$$y_2 = M(S_1S_2) = -0,07.$$

We remark that if it is necessary to continue the calculations, we have a choice of one of two procedures:

(a) Take from the beginning the main rhythm h large enough to obtain $y_3, \ldots, y_k$ (where k is fixed in advance.)

(b) Instruct the machine to dilute automatically the spectra to the greater rhythms which will be compatible with the next calculation.

The first procedure is ordinarily better if k is a small number.

It is possible to use spectra for solving difference equations even when the data are approximate numbers (interval numbers). Methods of solving difference equations having interval numbers, without the use of spectra are given for instance in [12] and [6].

7.2. APPLICATION OF SPECTRA TO BERNOULLI'S METHOD. As another example of the use of numerical spectra in recursive calculations, we consider Bernoulli's method for solving polynomial equations. We write the polynomial equation in the form

$$x^n = \sum_{i=1}^{n} b_i x^{n-1} = Q(x).$$

The setting of the method is to calculate a sequence of numbers u_k by means of the following recursive formula

$$u_k = \sum_{i=1}^{n} b_i u_{k-i} \qquad k = 1,2,\ldots$$

(See, for instance, [1], [7], [10]).

The initial values $u_o, u_{-1}, \ldots, u_{-(n-1)}$ can be prescribed as desired, but usually they are chosen as

$$u_o = 1, \quad u_{-1} = u_{-2} = \ldots = u_{-(n-1)} = 0.$$

The numerical spectra can be successfully used for this calculation. The two sequences which were mentioned in Section 7.1 can be chosen to be

(I) $b_1 \quad b_2 \quad b_3 \quad \ldots \quad b_{n-1} \quad b_n$

(II) $u_{k-n} \quad u_{k+1-n} \quad u_{k+2-n} \ldots u_{k-2} \quad k-1$

Now we determine the rhythm. If all the b_i are integers, the rhythm valid for p consecutive calculations is

$$h = [\log 2(m\ u\ v)^p] + 1$$

where m is the number of b_i different from zero and $u = \max|b_i|$, $v = \max|\mu_k|$. The modification to noninteger b_i is simple.

It is obvious that

$$\mu_k = M(S_1\ S_2)$$

where S_1 and S_2 are the spectra of the sequences (I) and (II) respectively.

8. PSEUDOSPECTRA WITH APPLICATIONS TO DIFFERENTIAL EQUATIONS.

8.1. PSEUDOSPECTRA.

If a positive integer h is too small to be compatible with the polynomial or with the function, then it is impossible to form the spectrum even though it is possible to form by means of the formulae (2.6)-(2.9) the numbers S. But these numbers are surely not spectra. Such numbers are called pseudospectra. In the present section we establish the relevance of pseudospectra to numerical spectral analysis. In spectra, all the digits of a certain coefficient a_i of the polynomial are not interlaced with the digits of the other coefficients of the polynomial. In the proposed pseudospectra such a mixing will occur. The first and key step in any spectral calculation is the determination of the rhythm h compatible with the problem and the given data. This rhythm is determined using majorizations which sometimes lead to a very large h. The result of the calculations sometimes indicates that such calculations could have been performed with a smaller rhythm. Also, in some problems of analysis, the determination of the rhythm becomes impractical due to the complexity of the quantities to be majorized. The pseudospectra enables us to carry on spectral analysis without an apriori knowledge of the rhythm. We now present the basic idea underlying pseudospectral processes.

The pseudospectral method starts with two natural numbers h_1, h_2 ($h_2 \cdot h_1$). Using the fixed number h_1 and the appropriate formulae (2.6)-(2.9), we obtain numbers $S_1^{(1)}$, $S_2^{(1)}$, $S_3^{(1)}$, ... The superscript is intended to indicate that the computations are carried using the rhythm h_1, and the subscripts refer to different elements of the data. By performing the required operations (such operations were described abstractly in (3.2)) we obtain a number $S^{(1)}$ that can be a spectrum or a pseudospectrum. We proceed in the same manner with h_2, obtaining a

resulting number $S^{(2)}$. By comparing these two numbers using the comparison theorem to be given below we can determine if $S^{(1)}$ is a spectrum. If $S^{(1)}$ is a spectrum, we are done. If it is not a spectrum, we take another natural number $h_3 > h_2$, form $S^{(3)}$ and repeat the same comparison, now with $S^{(2)}$ and $S^{(3)}$. This process can be carried on until, by means of comparison of $S^{(k-1)}$ and $S^{(k)}$, $S^{(k-1)}$ is found to be a spectrum.

Now we come to the comparison theorem alluded to above, which can be established without much difficulty.

Comparison Theorem. A necessary condition for $S^{(1)}$ and $S^{(2)}$ to be spectra is that the effective value of each section of $S^{(1)}$ is equal to the effective value of the corresponding section of $S^{(2)}$. This condition is also sufficient if the unknowns of the problems to be found by these spectra are all positive numbers (+).

Corollary. If not all effective values of two pseudospectra but only the effective values of the first k sections are equal, then the pseudospectra can be considered as spectra in these sections only.

Remark: It may appear that the calculations of the numbers $S^{(1)}, S^{(2)}, \ldots, S^{(k-1)}, S^{(k)}$ require more operations than the calculations of one spectrum using a rhythm h obtained by majorization estimates. However, this is not necessarily true if the rhythm h is much larger than h_k.

8.2. A SPECTRAL METHOD FOR SOLUTION OF ORDINARY DIFFERENTIAL EQUATIONS OF THE FIRST ORDER.

In this section we develop spectral and pseudospectral methods to solve the initial value problem

$$y' = f(x,y),\ y(x_o) = y_o \qquad (8.1)$$

We assume that the function $f(x,y)$ is expandable in Taylor series at (x_o, y_o), i.e.,

$$f(x,y) = \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} A_{mn}(x-x_o)^n\ (y-y_o)^m$$

(+)If this is not the case, then additional conditions are required for the sufficiency. The formulation of these conditions can only be given for specific classes of problems.

The basis for the application of the pseudospectral method to this differential equation is a modified Picard's method. The modification was made by one of the authors of this paper [13]. Because this modification is not widely known, we explain it briefly. The method consists of finding a sequence of polynomials $y_1, y_2, \ldots$ by means of the iterative scheme

$$y_i(x) = y_o + \int_{x_o}^{x} T_i\, f(x, y_{i-1}(x))dx$$

where the operator T_i is the Taylor polynomial of degree i-1 associated with $f(x,y_{i-1})$ at x_o.

$$f(x,y_{i-1}) = a_o^{(i)} + a_1^{(i)}\,(x-x_o) + \ldots + a_{i-1}^{(i)}\,(x-x_o)^{i-1}$$

It was shown in [13] rhat $a_k^{(i)}$ is independent of i for each fixed k, so the ith approximation consists of adding one new terms of the form $\int_{x_o}^{x} a_{i-1}(x-x_o)^{i-1}dx$. Furthermore, the solution of (8.1) (cf. [13])is given by

$$y_o + \sum_{j=0}^{\infty} \frac{a_j}{(j+1)!}\,(x - x_o)^{j+1}. \qquad (8.2)$$

For application of spectral or pseudospectral methods, it is necessary to know in advance that the coefficients a_j are integers or finite decimal numbers. For simplicity of exposition, we assume that all a_j are integers; the modification is similar to previous modifications elaborated in this paper. These conditions are satisfied by a large class of functions, for instance, the rational function

$$f(x,y) = \frac{P(x,y)}{Q(x,y)} \qquad (8.3)$$

satisfies these conditions if the coefficients of the polynomials P and Q are integers and $Q(x_o,y_o) = 1$.

In the iteration method exposed above there occur three kinds of functions. We construct from them the following table

$$
\begin{array}{lll}
 & & y_o \\
f(x,y_o) & T_1 f(x,y_o) & y_1 \\
f(x,y_1) & T_2 f(x,y_o) & y_2 \\
\cdots & \cdots & \cdots
\end{array}
\qquad (8.4)
$$

The elements of the first column are expandable in Taylor series, the elements of the second and third columns are polynomials. From the table (8.4), composed of functions, we form the table of corresponding numbers, spectra or pseudospectra, which we shall write in the following way:

$$
\begin{array}{lll}
 & & s_o \\
S_1 & \tilde{S}_1 & s_1 \\
S_2 & \tilde{S}_2 & s_2 \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot
\end{array}
\qquad (8.5)
$$

We note that the s_i cannot be spectra because the functions y_i in general have coefficients which are not finite decimal numbers. Thus the s_i are pseudospectra. From the relation

$$s_i = f(10^{-h}, s_{i-1}) \qquad (8.6)$$

It follows that the S_i must also be pseudospectra. The terms of the second column are obviously obtained from the corresponding terms of the first column of the same table (8.5) by rounding off and consequently are designated by the symbol ~ having the meaning of ablation of decimals.

Now suppose we want to find the first p terms of the series (8.2). This means that we have to obtain the $(p-1)^{th}$ successive approximation

$$y_{p-1} = \sum_{j=0}^{p-2} \frac{a_j}{(j+1)!} (x-x_o)^{j+1} \qquad (8.7)$$

But instead of finding y_{p-1} we shall find the polynomial $(p-1)!y_{p-1}$ since the coefficients of the latter polynomial are all integers. We modify tables (8.4) and (8.5) accordingly. The modification of (8.5) is given below, where we have also augmented the table by a new column of integers a_j.

$$\begin{array}{cccc} & & y_o & s_o \\ S_1' & \tilde{S}_1' & a_o & s_1 \\ S_2' & \tilde{S}_2' & a_1 & s_2 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{array} \qquad (8.8)$$

Here $S_i' = (i-1)!\ S_i$ and $\tilde{S}_i'$ is the value of S_i' rounded off to $(i-1)h$ decimals.

Now it remains only to find the relations by means of which the recursive calculations of the table (8.8) can be performed in a purely arithmetic way, row by row. It is easy to show that these relations are

$$\begin{aligned} S_i' &= (i-1)'\ f(10^{-h}s_{i-1}) \\ a_i &= 10^{ih}(\tilde{S}_{i+1}' - i\ \tilde{S}_i') \\ s_{i+1} &= s_i + \frac{10^{-h}}{(i+1)!}\ (\tilde{S}_{i+1}' - \tilde{S}_i'), \quad i = 1,2,\ldots \end{aligned} \qquad (8.9)$$

The last relation is obtained from

$$y_{i+1} = y_i + \frac{a_i}{(i+1)!}\ (x-x_o)^{i+1}$$

The initial values necessary to start the iterative processes are

$$s_o = y_o,\ s_1 = s_o + 10^{-h}\ \tilde{S}_1',\quad a_o = \tilde{S}_1' \qquad (8.10)$$

We remark that the numbers S_i' need only be calculated to the accuracy of $(i-1)h$ decimals; this approximate value is just $\tilde{S}_i$.

The calculations by means of formulae (8.9) can be performed using spectra or pseudospectra. For spectral processes it is necessary to have some knowledge about the coefficients A_{mn} of the Taylor series

expansion of the function $f(x,y)$. Also the rhythm for spectra which may be obtained is ordinarily very large. For these two reasons we shall adopt pseudospectral processes in which we do not need to know the coefficients A_{mn} and we can operate with smaller rhythms. The formulae for pseudospectral calculations are the same, but every number S_1' shall be calculated at least in two ways, by means of two arbitrary rhythms h_1 and h_2 $(h_2 > h_1)$. If the comparison of these two numbers $S_i'^{(1)}$, $S_i'^{(2)}$ leads to the same value of a_{i-1}, then we pass to the calculation of the next row. If the comparison leads to different a_{i-1} we take greater rhythms h_1' and h_2' and repeat the calculation. The chance of making a mistake (because the conditions of the general comparison theorem are not sufficient) is very small. Even if it occurs, it is detected by checking up of the final result, that is of the approximate solution of the differential equation.

Example: $$y' = \frac{1-x+y}{1-x^2 y} \qquad x_o = 0, \quad y_o = 0$$

It is necessary to calculate the coefficients a_i, $i = 0,1,2,3,4$. We take $h_1 = 1$, $h_2 = 2$. Table (8.8) (with the first column omitted) is in this case

	$y_o = a$	$s_o = 0$
$\tilde{S}_1'^{(1)} = 1$	$a_o = 1$	$s_1^{(1)} = 0.1$
$\tilde{S}_1'^{(2)} = 1$		$s_1^{(2)} = 0.01$
$\tilde{S}_2'^{(1)} = 1.0$		$s_2^{(1)} = 0.10$
$\tilde{S}_2'^{(2)} = 1.00$	$a_1 = 0$	$s_2^{(2)} = 0.0100$
$\tilde{S}_3'^{(1)} = 2.00$	$a_2 = 0$	$s_3^{(1)} = 0.100$
$\tilde{S}_3'^{(2)} = 2.0000$		$s_3^{(2)} = 0.010000$
$\tilde{S}_4'^{(1)} = 5.994$	$a_3 = -6$	$s_4^{(1)} = 0.099975$
$\tilde{S}_4'^{(2)} = 5.99999$		$s_4^{(2)} = 0.009999975$
$\tilde{S}_5'^{(1)} = 23.9754$		
$\tilde{S}_5'^{(2)} = 23.99997594$	$a_4 = -6$	

Thus we obtain $y = x - \frac{x^4}{4} + \frac{x^5}{20} + \ldots$

The spectral and pseudospectral methods can be used for differential equations of higher order and for systems of differential equations.

REFERENCES

1. Aitken, A. C., On Bernoulli's numerical solution of algebraic equations. Proc. Roy. Soc. Edinburgh 46 (1926), 289-305.

2. Bareiss, E. H., Resultant procedure and the mechanization of the Graeffe process, J. Assoc. Comp. Mach. 7 (1960), 346-386.

3. Conte, S. D., Elementary Numerical Analysis: An Algorithmic Approach. McGraw-Hill, New York, 1965.

4. Faddeev, D. K., and Faddeeva, V. N., Computational Methods of Linear Algebra (translated from Russian by R. C. Williams), Freeman, San Francisco, 1963.

5. Graeffe, C. H., Die Auflösung der höhren numerischen Gleichungen, als Beantwortung einer von der koniglichen Akademie der Weissenschaften zu Berlin aufgestellten Preisfrage, pp. 1 - 44. Friedrich Schulthess, Zurich, 1937.

6. Greenspan, D., Introduction to Calculus, Harper and Row, New York, 1968.

7. Henrici, P., Elements of Numerical Analysis. Wiley, New York, 1964.

8. Householder, A. C., Principles of Numerical Analysis. McGraw-Hill, New York, 1953.

9. Lorch, E. R., Spectral Theory. Oxford University Press, New York, 1961.

10. Mihajlović, B., and Pecka, F. Primena matematickih spektara na metodu Bernoulli-a za iznalaženje korena algebraske. jednačine. Vesnik Društva matem. i fiz. 3 (1964), 187-191.

11. Moore, R. E., The automatic analysis and control of error in digital computing based on interval arithmetic. In L. B. Rall (ed.), Error in Digital Computation, Vol. 1, Wiley, New York, 1965, pp. 61-130.

12. Moore, R. E., Interval Analysis. Prentice-Hall, Englewood Cliffs, New Jersey, 1966.

13. Orloff, C., Un procédé d'approximation concernant les integrales des équations différentielle. Bull. de l'Academie Serbe 2 (1935), Belgrade. See also Orlov, K, Jedna metoda aproksimiranja za integrale diferencijalnih jednačina. Glas Srp. Akad. Nauka CLXIII, prvi razred 80 (1934).

14. Orloff, C., Méthode spectrale pratique d'evaluation numerique des determinants et de résolution du système d'équations linéaires. Vesnik Društva matem. i fiz, srbije 5 (1953), 17-30.

15. Orloff, C., Application pratique de la théorie des spectres mathématiques de Michel Petrovitch au Calcul numérique. La Revue Scientifique (Paris), 91 Annee (1953), 243-247.

16. Orloff, C., Sur la méthode de Graeffe. Comptes Rendus de l'Acad. de Sciences (Paris), 243 (1956), 1269-1270.

17. Orloff, C., Application des spectres mathematiques à la résolution des équations différentielles ordinaires. Bull. Soc. Math. Phys. Serbie (Belgrade) 9 (1957), 283-294

18. Orlov, K., Nouvelle méthode spectrale de résolution des equations algebiques. Matematički Vesnik (Belgrade) 3 (18)(1966), 287-296.

19. Ostrowski, A., Recherches sur la méthode de Graeffe et les zeros des polynomes et les séries de Laurent. Acta Math. 72(1940), 99-257.

20. Petrovitch, M., Les spectres numériques. Gauthier-Villars, Paris, 1919.

21. Petrovitch, M., Lecons sur les spectres mathematiques professées la Sorbornne en 1928. Gauthier-Villars, Paris, 1928.

22. Rall, L., Computational Solution of Nonlinear Operator Equations. Wiley, New York, 1969.

23. Ralston, A., A First Course in Numerical Analysis. McGraw-Hill, New York, 1965.

24. Todd, J. (ed.), Survey of Numerical Analysis. McGraw-Hill, New York, 1962.

LIST OF ATTENDEES

1971 ARMY NUMERICAL ANALYSIS CONFERENCE

1. Mr. James A. Ary
   AF/SAAD Studies Analysis
   1119 19 Street
   Rosslynn, Virginia 22209

2. Mr. John A. Battilega
   US Army Strategy & Tactics Analysis Group
   8120 Woodmont Avenue
   Bethesda, Maryland 20014

3. Mr. Jesse Bemley
   US Army Topographic Command
   Washington, D. C. 20315

4. Dr. Jenny Bramley
   Department of the Army
   US Army Engineer Topographic Labs
   Fort Belvoir, Virginia 22060

5. [illegible]verett C. Brown
   [illegible]onal Civil Defense Computer Facility
   [illegible]ney, Maryland 20832

6. Mr. Joseph J. Budelis
   US Army Natick Labs
   Natick, Massachusetts 01760

7. Dr. E. F. Caviness
   Mathematics Research Center
   University of Wisconsin
   Madison, Wisconsin 53706

8. Dr. Jagdish Chandra
   Mathematics Division
   US Army Research Office-Durham
   Box CM, Duke Station
   Durham, North Carolina 27706

9. Mr. James B. Cheek, Jr.
   P. O. Box 631
   US Army Engineer Waterways Experiment Station
   Corps of Engineers
   Vicksburg, Mississippi 39180

10. Dr. C. Y. Cho
US.Army Advanced Materiel Concepts Agency
Washington, D. C. 20315

11. Mrs. Bertha Cory
US Army Behavior & Systems Research Lab
1300 Wilson Boulevard
Arlington, Virginia 22209

12. Mr. R. Crenwel, Jr.
US Army Topographic Command
Washington, D. C. 20315

13. Dr. F. G. Dressel
US Army Research Office-Durham
Box CM, Duke Station
Durham, North Carolina 27706

14. Mr. David Dubois
US Army Aeromedical Research Lab
Fort Rucker, Alabama 36360

15. Mr. S. H. Eisman
L6200 BLDG. 513
Frankford Arsenal
Philadelphia, Pennsylvania 19137

16. Mr. Isaac Fail
US Army Topographic Command
Washington, D. C. 20315

17. CPT Clifford W. Greve
Department of the Army
US Army Engineer Topographic Labs
Fort Belvoir, Virginia 22060

18. Dr. John H. Giese
Chief, Applied Mathematics Division
Aberdeen Research & Development Center
Aberdeen Proving Ground, Maryland 21005

19. Dr. James H. Griesmer
College of Engineering
Department of Electrical Engineering & Computer Sciences
University of California
Berkeley, California 94720

20. Mr. Sol Haberman
The Johns Hopkins University
Applied Physics Lab
8621 Georgia Avenue
Silver Spring, Maryland 20910

21. Mr. Arthur Hausner
Harry Diamond Laboratories
Washington, D. C. 20438

22. Dr. F. Heinmets
Head, Biophysics Group
Pioneering Research Lab
US Army Natick Laboratories
Natick, Massachusetts 01766

23. Mr. P. Hershall
Harry Diamond Laboratories
Washington, D. C. 20438

24. Mr. E. Hoffman
US Army Topographic Command
Washington, D. C. 20315

25. Mr. R. E. Hopkins
US Army Mobility Equipment R & D Center
Fort Belvoir, Virginia 22060

26. Mr. David Howes
US Army Strategy & Tactics Analysis Group
8120 Woodmont Avenue
Bethesda, Maryland 20014

27. Dr. H. S. Hung
Mathematics Research Center
University of Wisconsin
Madison, Wisconsin 53706

28. Mr. M. A. Hussain
Maggs Research Center
Watervliet Arsenal
Watervliet, New York 12189

29. Mr. James F. Jacobs
National Civil Defense Computer Facility
Olney, Maryland 20832

30. Mr. M. Kaplan
US Army Topographic Command
Washington, D. C. 20315

31. Mr. Donald F. Kass
US Army Natick Labs
Natick, Massachusetts 01760

32. Mr. George Keiser
US Army Aeromedical Research Labs
Box 577
Fort Rucker, Alabama 36360

33. Mr. A. R. Kiwan
Ballistic Research Lab
Aberdeen Proving Ground, Maryland 21005

34. Mr. Paul O. Langgoth
US Army Materiel Command
AMCRD-ES
BLDG. T-7
Washington, D. C. 20315

35. Dr. Siegfried Lehnigk
Physical Sciences Laboratory
US Army Missile Command
Redstone Arsenal, Alabama 35809

36. Mr. Philip G. Lem
CSL-ETL
Fort Belvoir, Virginia 22060

37. Mr. Andrew A. Lesick
New London Lab
Naval Underwater Systems Center
New London, Connecticut 06320

38. Mr. Leon Leskowitz
US Army Electronics R&D Command
Computation Center, Mathematics Division
AMSEL-GG-M
Fort Monmouth, New Jersey 07703

39. Mr. V. Lewicke
US Army Topographic Command
Washington, D. C. 20315

40. Professor J. C. R. Licklider
Director, Project MAC
545 Technology Square
Cambridge, Massachusetts 02139

41. Mr. Roger MacGowan
Department of Defense Computer Institute
Washington Navy Yard
Washington, D. C. 20390

42. Mr. Joseph Mandelson
Chief, Quality Evaluation Division
Quality Assurance Directorate
ATTN: SMUEA-QA-A
Edgewood Arsenal, Maryland 21010

43. Mr. Robert A. McMurrer, Chief
Systems Support Division
Engineer Information & Data Systems Office
Office of the Chief of Engineers
Washington, D. C. 20314

44. CPT Isaac S. Metts, Jr.
Walter Reed Army Institute of Research
Walter Reed Army Medical Center
Washington, D. C. 20012

45. COL M. Mosteller
OPNAV (96L5)
Amphibious Warfare
Department of the Navy
Washington, D. C.

46. Dr. M. Z. Nashed
Mathematics Research Center
University of Wisconsin
Madison, Wisconsin 53706

47. Mr. A. C. Noma
US Army Topographic Command
Washington, D. C. 20315

48. Miss Mary Rita Powers
Naval Underwater Systems Command
New London, Connecticut 06320

49. Dr. Louis B. Rall
Mathematics Research Center
University of Wisconsin
Madison, Wisconsin 53706

50. Mr. W. B. Riggins, Jr.
D/CS
US Army Topographic Command
Washington, D. C. 20315

51. Dr. Stephen M. Robinson
Mathematics Research Center
University of Wisconsin
Madison, Wisconsin 53706

52. Dr. Ed Runnion
School of Logistic Sciences
US Army Logistics Management Center
Fort Lee, Virginia 23801

53. Mr. William Sacco
Ballistic Research Lab
Aberdeen Proving Ground, Maryland 21005

54. Mr. R. D. Scanlon
Maggs Research Center
US Army Watervliet Arsenal
Watervliet, New York 12189

55. Mr. Gerald M. Schultz
Harry Diamond Labs
Washington, D. C. 20438

56. Mr. R. J. Sekel
US Army
OCE
Washington, D. C.

57. Mr. Howard Selkin
US Army Topographic Command
Washington, D. C. 20315

58. Mr. Ralph Sellers
US Army Missile Command
Redstone Arsenal, Alabama 35809

59. MAJ Clay Smith
Office of Chief of Research & Development
Washington, D. C.

60. Mr. Knute J. Takle
Code MAT 0335
Department of the Navy
Washington, D. C. 20360

61. Mr. G. Taylor
US Army Topographic Command
Washington, D. C. 20315

62. LT Dan Troyan
Department of Defense Computer Institute
Washington Navy Yard
Washington, D. C. 20390

63. Mr. Joseph Tylor
Applied Mathematics Branch
Systems Analysis Division
Edgewood Arsenal, Maryland 21010

64. Professor A. Van Dam
Computing Center
Brown University
Providence, Rhode Island 02912

65. Foster Walker, Jr.
HQ, US Army Materiel Command
Washington, D. C. 20315

66. Dr. James A. Ward
Naval Ordnance Systems Command
National Center #2, Rm. 10E92
ATTN: Code 6204E
Washington, D. C. 20360

67. Dr. J. Michael Yohe
Mathematics Research Center
University of Wisconsin
Madison, Wisconsin 53706

68. Mr. C. K. Zoltani
Ballistic Research Lab
Aberdeen Proving Ground, Maryland 21005

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
U. S. Army Research Office-Durham Box CM, Duke Station, Durham, North Carolina 27706	Unclassified
	2b. GROUP N/a

3. REPORT TITLE

PROCEEDINGS OF THE 1971 ARMY NUMERICAL ANALYSIS CONFERENCE

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Interim Technical Report

5. AUTHOR(S) (First name, middle initial, last name)

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
December 1971	279	

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	ARO-D Report 71-4
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited. The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
None	Army Mathematics Steering Committee on behalf of the Office of the Chief of Research and Development

13. ABSTRACT

This is the technical report resulting from the 1971 Army Numerical Analysis Conference. It contains papers on computer aided design and engineering, as well as papers on numerical analysis.

14. Key Words:

scratchpad/1
symbolic mathematics
TV techniques applied to numerical computations
computer simulation of biological problems
multiple regression equations
nonlinear programming
purturbed linear systems
finite element analysis
spline interpolation methods
iterative solutions
abstract polynomials
rounding
numerical spectra

DD FORM 1 NOV 68 1473 REPLACES DD FORM 1473, 1 JAN 64, WHICH IS OBSOLETE FOR ARMY USE.

Unclassified

Security Classification