EUR 291.e
REPRINT

EUROPEAN ATOMIC ENERGY COMMUNITY - EURATOM

APACHE A BREAKTHROUGH IN ANALOG COMPUTING

by

A. DEBROUX - H. D'HOOP - C. GREEN

1963

Joint Nuclear Research Centre
Ispra Establishment (Italy)
Scientific Data Processing Centre (CETIS)

Reprinted from
IRE TRANSACTIONS ON ELECTRONIC COMPUTERS
Vol. EC-11, No. 5 - 1962

This reprint intended for restricted distribution only. It reproduces, by kind permission of the publisher, an article from „IRE TRANSACTIONS ON ELECTRONIC COMPUTERS", Vol. EC-11, No. 5 - 1962, 699-706. For further copies please apply to IRE - Professional Group on Electronic Computers, 1 East 79th Street — New York 21, N.Y., U.S.A.

Dieser Sonderdruck ist für eine beschränkte Verteilung bestimmt. Die Wiedergabe des vorliegenden in „IRE TRANSACTIONS ON ELECTRONIC COMPUTERS", Vol. EC-11, Nr. 5 - 1962, 699-706 erschienenen Aufsatzes erfolgt mit freundlicher Genehmigung des Herausgebers. Bestellungen weiterer Exemplare sind an IRE - Professional Group on Electronic Computers, 1 East 79th Street — New York 21, N.Y., USA., zu richten

Ce tiré-à-part est exclusivement destiné à une diffusion restreinte. Il reprend, avec l'aimable autorisation de l'éditeur, un article publié dans «IRE TRANSACTIONS ON ELECTRONIC COMPUTERS», Vol. EC-11, N° 5 - 1962, 699-706. Tout autre exemplaire de cet article doit être demandé à IRE - Professional Group on Electronic Computers, 1 East 79 th Street — New York 21, N.Y., U.S.A.

Questo estratto e destinato esclusivamente ad una diffusione limitata. Esso è stato riprodotto, per gentile concessione dell'Editore, da «IRE TRANSACTIONS ON ELECTRONIC COMPUTERS», Vol. EC-11, N° 5 - 1962, 699-706. Ulteriori copie dell'articolo debbono essere richieste a IRE - Professional Group on Electronic Computers, 1 East 79th Street — New York 21, N.Y., U.S.A.

Deze overdruk is slechts voor beperkte verspreiding bestemd. Het artikel is met welwillende toestemming van de uitgever overgenomen uit „IRE TRANSACTIONS ON ELECTRONIC COMPUTERS", Vol. EC-11, no. 5 - 1962, 699-706. Meer exemplaren kunnen besteld worden bij IRE - Professional Group on Electronic Computers, 1 East 79th Street — New York 21, N.Y., U.S.A.

Reprinted from IRE TRANSACTIONS
ON *ELECTRONIC COMPUTERS*
Volume EC-11, Number 5, October, 1962

PRINTED IN THE U.S.A.

APACHE—A Breakthrough in Analog Computing*

C. GREEN†, H. D'HOOP†, AND A. DEBROUX†

Summary—**This paper describes the working of the APACHE (Analog Programing And CHEcking) system for the production of analog computer programs using a digital computer, which has been developed jointly by digital and analog programers at the computation center of EURATOM in Ispra, Italy. The language which has been formulated for the writing of analog problems is discussed and its rules are set out in the Appendix, while the processes involved in the utilization of the program are described and illustrated by an example of the listing of the digital computer input and output for a range of equation types. Finally, the significance of this new programing system in terms of economic exploitation of the analog machine, the use of trained personnel, and the design of new computers, are considered briefly.**

Introduction

IF ONE CONSIDERS the advantages and disadvantages of the analog method for the solution of engineering and scientific problems, one finds on the debit side two items of outstanding importance. The first of these is the cost of preparing a problem to the point at which useful results are produced, in terms of both engineering and machine time, not forgetting that for the analog machine the bulk of this price must be paid *each time* a given problem is set up. The second is the great difficulty of ensuring that the given equations are being solved at all times during the production of results. It is worth remarking that the cost of a minor error is proportional to the time which elapses before it is discovered, and therefore, the cost can be very high.

There are cases in which engineers studying large systems of a type which one would suppose ideal for the analog treatment have turned to digital simulation for these economic reasons. With the steady improvements in digital machines and programing techniques this trend could become very important.

On the other hand, once the problem is posed, the solution becomes apparent. The question at issue is whether the process followed by the analog programer can be treated as a problem in data reduction according to defined rules in order that the programing and checking process may be handled automatically; the APACHE system gives an affirmative answer to this question.

APACHE (Analog Programing And CHEcking) is the result of a combined study by digital and analog specialists at the Calculation Center of EURATOM, Ispra, (Italy)[1] and consists of a digital computer program which accepts the essential problem data in a raw state and produces the analog program in such a form as to make maximum use of automation and logical feedback possibilities.

Great emphasis has been placed on the *language* employed, in the hope that if a language is found to be logical, clear and concise, it will also be of permanent and universal application.[2]

Thus, the rest of this article is devoted to a description of this language and program, including its exploi-

* Received February 19, 1962; revised manuscript received, March 10, 1962.

† EURATOM Computation Center, Ispra, Italy.

[1] The EURATOM Computation Center is equipped with a 250 amplifier, PACE 231R analog installation and a 7090 digital computer. The APACHE code uses 8 tape units, is run by a FORTRAN monitor system; it makes use of indirect addressing and of the "Convert" instructions.

[2] A detailed account of the language will be found in communication No. 107 presented at the 3rd Internatl. Conf. on Analogue Computation, Opatija, Yugoslavia; September, 1961.

tation, and it concludes with an assessment of future development, in the course of which it will be shown that the APACHE system offers considerable savings of money to owners of both large and small analog installations and also provides the most powerful economic justification of remote controlled setting-up facilities for the analog computer.

Writing Input Data

The criteria adopted in the choice of form for input to program were:

1) Economy in writing,
2) Correspondence with the best basic possible form of mathematical symbolism in general use,
3) Uniqueness in interpretation at the stage of analog set-up,
4) Flexibility allowing the possibility of writing a problem to produce any desired computer set-up.

The reader is invited to examine the programing phase of an analog problem using APACHE, at the stage when the analyst is able to write a mathematical model according to the following form:

1) a set of differential equations

$$\frac{d^n x_i}{dt^n} = f_i\left(\frac{d^{n-1}(x_i)}{dt^{n-i}} \cdots \frac{d(x_i)}{dt}, X_1 \cdots X_i \cdots X_m, Y_1 \cdots Y_n, Z \cdot Z_n\right) \qquad i \text{ from } 1 \text{ to } m,$$

2) a set of explicit algebraic equations

$$y_j = f_j(X_1 \cdots X_m, Y_1 \cdots Y_j \cdots Y_p, Z_1 \cdots Z_q) \qquad j \text{ from } 1 \text{ to } p,$$

3) a set of implicit relations (geometrical constraints, etc., $\cdots$)

$$0 = f_k(Z_i \cdots Z_k \cdots Z_q) \qquad k \text{ from } 1 \text{ to } q,$$

4) a set of arbitrary functions to feed function generators, coordinate transformations and logical conditions necessitating relays.

Where differential equations require integrators, explicit algebraic equations summers and the implicit relations are taken care of by means of high-gain amplifiers. At this stage, a certain number of punched cards will be prepared, and the evident purpose of the language is to have an easy correspondence between usual mathematical notations and the characters on the keyboard of the card punch machine. So, the statements for programing a certain problem would have the following appearance:

PARAMETERS
$A = 1.0500$
and, in general:
alphanumerical expression = numerical value or parametric expression,

VARIABLES
$X1 = 50.00, 100.00$
and in general:
alphanumerical expression = initial value, maximum expected value.

EQUATIONS
$A * \text{DER2}\ (X1) = B * \text{DER1}\ (X1) + C * X2$
$F * \text{DER1}\ (X3) = Y1 * Y2 * (D + Z1/Z2).$

The words such as: PARAMETERS, VARIABLES or EQUATIONS are orders to the digital computer, determining how the data following will be organized in tables.

The familiar square-root arrangement using a high-gain amplifier will show how implicit relations are introduced.

The expression $y = \sqrt{x}$ is converted to $o = y^2 - x$ and in APACHE notations:

```
EQUATIONS
. . . . . . . . . . . . . . . .
ZERO (Y)=Y * *2-X
. . . . . . . . . . . . . . . .
```

where the expression ZERO (Y) indicates that Y is desired as the output of the high-gain amplifier, the right-hand terms being the inputs, as in all the other APACHE notations.

Another example (Fig. 1) shows the writing of a large number of equation types, and the rules for writing the input will be found in detail in the Appendix. The reader is invited, however, to judge the clarity of the language from the point of view of an engineer rather than a computation specialist. Although the whole process is implicit in the Appendix, a few generalities may be permitted here. The reader will observe that although an effort was made to keep the language strictly problem-oriented this soon proved impossible; concessions were made to analog technique, for example, in the treatment of resolutions of coordinates, as being carried out in a "black box," which is logically equivalent to a resolver.

A substantial advance with respect to the first two criteria above has been obtained by including a large part of the reduction of differential equations to first order in the automatic section of the processing.

This elimination of a manual transformation is important as it eliminates errors which *cannot* be checked at the stage of problem checkout. In a similar fashion, provision is made for the insertion of minimum as well as maximum values of variables, where appropriate, in which case the equations will automatically be transformed to work with perturbations about the average of these two values.

```
COMPILATION APACHE 2—DEMONSTRATION OF APACHE CODE NOMENCLATURE
15 SEPTEMBER 1961
PROGRAMMED BY          C. GREEN
PROBLEM NO.            132

INPUT LISTING FORMAT

 1  PARAMETERS
 2  A21        = 8.0
 3  A11        = 1.0
 4  A01        = 90
 5  B2         = 1.3333
 6  A1         = A2(1)
 7  B1         = B2(1)
 8  A31        = 0.03
 9  A34        = 0.2
10  A32        = 0.1
11  A33        =10.5
12  H          =1.32E4
13  C          =91.18
14  K          =100
15  A5         =22.8
16  K2         =1.8E-2
17  C41        =1.8
18  C42        = 9.9
19  K1         = 1.2
20  C2         = 1.3E2
21  K4         =1E2
22  BETA       = 3
23  VARIABLES
24  X1         = -11       , 15
25  Y1         = +Y1B(1)   , 50
26  X2         =           , 5        ,LIMITS,+5,-3
27  X3         = +2.6      , 3
28  Y2         = +1.1E2    , 2E2
29  X4         =-3.22      , 5
30  Y3         = -30       , -25      , -35
31  X1/Y2      =           , 1
32  DER1(X1)   = -9        , 15
33  DER2(X1)   = +31       , 45       ,TEST
34  F1Y2       = +11       , 12
35  FY2Y3      = +10       , 50
36  Y1/X1      =           , 3.5
37  (Y2-Y3)    =           , 125
38  DER2(Y2)   = +70       , 100      ,TEST
39  DER1(Y2)   = -40       , 50
40  SQR1       =           , 55
41  SIN1       =           , 3
42  COS1       =           , 18
43  SQR2       =           , 20
44  COS2       =           , 15
45  ARC1       =           , 0.4
46  F          =           , 22
47  SQR3       =           , 3E2
48  Q          =           , 125      ,EXACT
49  Q1         =           , 125      ,EXACT
50  Q2         =           , 0.5
51  REF3       =           , 100
52  REF4       =           , 100
53  REF1       =           , 100
54  REF2       =           , 100
55  EQUATIONS
56  DER(Y1)   = F1Y2+A31*X2*(SMS)Y2-A34*FY2Y3+A32*Y2-A33+B1/(A1+B1)*F
57  DER3(X1)  = -A21*DER2(Y2)-A11*DER1(X1)+A01*X1*(TDM)X2/(TDM)Y2-A01*B2*X1/(TDM)Y2+A1/(A1+B1)*F
58  ZERO(F)   = X1*(QSQ)Y1+H/C*(X1+Y1)-H*H
59  DER2(Y2)  = SQR1+K*Y1/(QSQ)X1+SIN1+COS 1
60  RES,RECTANGULAR,COS2,      =X1 ,X3
61  DER1(X4)  =Q+C41*REF3+C42*REF4
62  COMPARE(N2-X2)  REF3 ,REF4  =  REF
63  Q1        = (X4*(SMN)X4)*(SMN)X4
64  Q2        = -K1*ARC1
65  DER1(X3)  = 1*REF1+2REF2+C2*Q3
66  COMPARE(X3-M3)  Q=Q1 ,Q2
67  SWITCH,0,REF1,REF2,   =  REF
68  SWITCH,Q3  =  0 ,0 ,(Y2-Y3)
69  PPD(FY2Y3)  =  (FUNCTION,Y3)*(SMN)Y2
70  DFG10(FIY2)  =  Y2
71  DO 77  I = 1 ,5
72  DO 77  J = 1 ,7
73  DO 77  K = 1 ,3
74  COMPUTE
75  A1         = A2(I)
76  B1         = B2(J)
77  Y1A        = Y1B(K)
```

Fig. 1—Demonstration of APACHE code nomenclature. (*Cont'd. next page.*)

```
   PARAMETERS
78 A2(1)  =  1.5
79 A2(2)  =  1.4
80 A2(3)  =  1.3
81 A2(4)  =  1.2
82 A2(5)  =1.0
83 B2(1)  =  0.03
84 B2(2)  =  0.04
85 B2(3)  =  0.05
86 B2(4)  =  0.06
87 B2(5)  =  0.07
88 B2(6)  =  0.08
89 B2(7)  =  0.1
90 Y1B(1) =  40
91 Y1B(2) =  50
92 Y1B(3) =  60
   END

COMMENTS

OUTPUT LISTING FORMAT SAMPLE. DEVELOPMENT OF COMPLEMENTARY EQUATIONS
  57.0  DER(DER2(X1)) = -A21 * DER2(Y2)-A11 * DER1(X1)+A01 * (X1 * X2/Y2)-A01 * B2 * (X1/Y2)+A1/(A1+B1) * F
  57.1  DER(DER1(X1)) = +DER2(X1)
  57.2  DER(X1)       = +DER1(X1)
  57.3  (X1 * X2/Y2)  = +X1 * X2/Y2
  57.4  (X1/Y2)       = +X1/Y2

COMMENTS

OUTPUT LISTING FORMAT SAMPLE. COMPLETE DEVELOPMENT OF AN EQUATION
  56.01 DER(Y1)             = +            F1Y2      +A31      * (X2 * Y2)        -A34     * FY2Y3     +A32     * Y2
  56.02 5.7779              = +11.000                +(-3.5947)                   -2.0000              +11.000
  56.03 DERTAU(2 * Y1) (-1)= -0.25/B   * (8 * F1Y2)-0.6/B    * (0-1 * (X2 * Y2))+0.2/B   * (2 * FY2Y3)-0.4/B   * (0.5 * Y2)
  56.04 +03.85         (-1)= -07.33                 +02.40                       +01.33               -07.33
  56.05 +03.85         (-1)= -1 * .0833 * (+88.00) -1 * .2000 * (-11.98)        +1 * .0667 * (+20.00) -1 * .1333 * (+55.00)

  56.06 INT A31 (A42)  (-1)= +1 *  Q30  DFG F62 +1 * Q80   A66                 +1 *  P32   A32        +1 * P33   A33
  56.07 A34            (-1)= +1 *  A31

  56.01                      -A33           +B1/(A1+B1) * F
  56.02                      -10.500        +(-0.1274)
  56.03                      +0.21/B  * REF -00.098/B    * (4 * F)
  56.04                      +07.00         +00.08
  56.05                      +1 * .0700 * (100) -1 * .0033   * (-26.00)

  56.06                      +1 * P33       +1 * P24      A24

  56.11 (X2 * Y2)           = +X2          * Y2
  56.12 -119.81             = +3.9941      * (-30)
  56.13 (0.1(X2 * Y2))      = +(20 * X2)/100 * (0.5 * Y2)
  56.14 -11.98              = -11.98
  56.15 -11.98              = +79.88/100   * (-15.00)
  56.16 SM1A                = +A12         (+A21-A33)
  56.17 A66            (-1)= +1 * SM1A
```

Fig. 1—*Cont'd.*

Use of Subroutines

By extension of a very common digital technique the notions of *subroutine* and *indexes* have been admitted, giving opportunities previously unknown to the analog programer. It will now be possible to preserve program sections in a permanent form, including everything up to the block diagram, without numbering, for which application should not be hard to find, especially in an installation such as that of EURATOM, which specializes in nuclear problems. For the writing of problems involving the finite difference technique, where one has a number of similar cells, the use of indexed variables effects a useful economy; but the main application of indexes will be in the automatic permutation of parameter and initial condition values resulting in a production potentiometer setting list or tape. The latter use is illustrated at the end of Fig. 1, and deserves notice as an example of a *sequential* operation.

Finally, it is to be noted that a sufficient set of orders has been provided to give the operator complete control over the block diagram produced and its allocation among the equipment available, which gives the opportunity to profit from machine checking of circuits designed and patched by hand, whereas the correction of a mistake in the writing of a problem is, in any case, immediate by virtue of the facility for the suppression and insertion of sections of program of any size.

Interpreting Output Data

After reduction of all introduced data, the digital computer will supply an output under two different forms:

1) A listing, written in a clear language,

2) A set of punched cards, coded.

The listing will be the equivalent of the "problem sheets" which are familiar to analog engineers. If the analog computer has only manual setting, this listing will be used to program the analog computer in the conventional way, since the following information will be displayed in successive rows:

1) The physical equation in its reduced form,
2) The same equation with the values of parameters and initial values of variables substituted, corresponding to the traditional static check calculation in physical units,
3) The scaled equation in its traditional form,
4) Gives the value in volts of each term, and corresponds to the outputs of the potentiometers,
5) Contains the coefficients to be set on the potentiometers together with the value in volts of the variable, the product of these terms being the value in line 4). These two lines, considered as equations, preserve their arithmetic sense.
6) The address line, written in correspondence with the scaled equation. It is to be noted that the flow of information is right to left, and the patching should proceed accordingly, as amplifier to potentiometer to stated gain of the amplifier on the left of the equality sign. This arrangement is repeated for each equation.

The reader will be able to verify that this layout makes clear the output in volts for every machine element for the checking stage, the terms having been carefully aligned for this purpose. The reason for the use of complementary equations also becomes clear at this point. The flow diagram is implicitly contained in the preceding information, but may be redrawn for clarity.

The output listing for the second-order differential equation given in the first example would have the following appearance:

```
1)  DER (DER1 (X1))                =-B/A       *DER1(X1)               +C/A       *X2
2)  -2.5000                        =-5.0000                            +2.5000
3)  -DER(2.0000 *DER(X1)) (-1)=-2.5000/B *(2.0000 *DER1 (X1)) +4.0000/B *(0.5000 *X2)
4)  +(05.00)                  (-1)=-(10.00)                            +(05.00)
5)  +(05.00)                  (-1)=-10 *.2500 *(04.00)                 +10 *.4000 *(0.125)
6)  A00 INT                        =-10 *Q00   *A00                    +10 *  Q02 *A13
```

It is worth remarking that experience suggests that the unification of all data relative to a given equation is greatly superior to the separate listing of each programing phase. However, tabulations of potentiometer values can be produced if required, and since nearly all analog machines have a wide variety of manual settings possible, for example function switches, positions of various types of multiplier and external switches governing the function of nonlinear units, a list of these settings appropriate to the machine concerned is included in the output listing.

It is to be noted, however, that everything up to address line is independent of the type of analog computer used, and in fact the circuit diagram exists in the digital computer in a general form. Thus, to adapt the program for another computer type requires little more than the rewriting of the allocation of addresses routine.

If the analog computer is equipped with digital input-output facilities and if pots are set by means of servos, (in our case we use an ADIOS coupled to a 026 IBM card reader) the system includes all coefficient settings and the selection and perforation of all voltages necessary to check the functioning of the analog computer and the problem itself. The perforations are then analyzed by the digital computer which puts out a complete list of faults. The automation also includes the preparation of the patch panel, for which instructions are decoded and fed to an optical apparatus, which will aid patching in such a way that this operation may be done by unskilled personnel.

Operating Procedures

There is a distinction to be drawn between the program stage and the problem running, in the sense that the first is available to all who have access to any digital facilities, while the second is a benefit reserved to the centers which have both digital and analog machines. It must be emphasized that the program stage of APACHE is completely self-contained and is the mechanized equivalent of the traditional form of analog program as produced at great cost by skilled engineers. Furthermore, the APACHE output is free from error.

In any case, the economic aspect is sufficient to justify the hiring of digital time, even at a distant location; for the preparation of program and checks and the procedure which is given here will not differ in principle whatever the degree of sophistication of the installation involved. For all users the existence of APACHE makes itself felt at an early stage in the problem acceptance process, since once the programer is satisfied with the representation of the problem given by the equations, and has decided how to treat points of special difficulty he can make a preliminary programing on the digital machine. All the consequences of the programing decisions taken can then be clearly seen and fed back into the problem appreciation stage. Thereafter, the complete program can be written out with care, the output listing is produced, the patch panel is prepared and placed on the machine, the function generators are set up and finally the nonlinear boxes and manual switches and the setting and checking programs are run.

In the case of a purely analog installation the process is then complete but the owner of a digital machine can still gain much from the system. His program is stored on magnetic tape for speedy access and when the output of the checking program is ready it is fed back into the digital computer for analysis and the resulting list of errors can be immediately corrected and checked. The general organization of the APACHE program will be found in Fig. 2.

It may be necessary to emphasize while the fully automatic compilation of perfect programs is admittedly impossible, the APACHE system is so designed as to give maximum assistance to all analog programers, both experts and beginners.

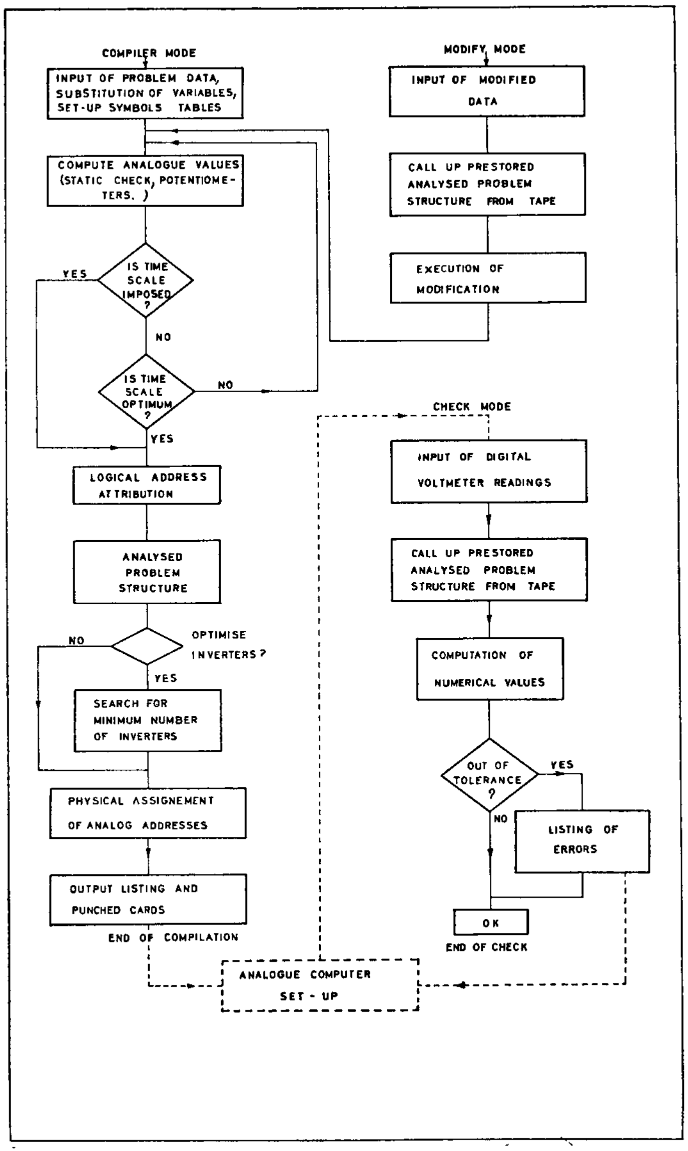
COMPILER MODE
INPUT OF PROBLEM DATA, SUBSTITUTION OF VARIABLES, SET-UP SYMBOLS TABLES
COMPUTE ANALOGUE VALUES (STATIC CHECK, POTENTIOMETERS.)
IS TIME SCALE IMPOSED ?
YES
NO
IS TIME SCALE OPTIMUM ?
NO
YES
LOGICAL ADDRESS ATTRIBUTION
ANALYSED PROBLEM STRUCTURE
OPTIMISE INVERTERS ?
NO
YES
SEARCH FOR MINIMUM NUMBER OF INVERTERS
PHYSICAL ASSIGNEMENT OF ANALOG ADDRESSES
OUTPUT LISTING AND PUNCHED CARDS
END OF COMPILATION
MODIFY MODE
INPUT OF MODIFIED DATA
CALL UP PRESTORED ANALYSED PROBLEM STRUCTURE FROM TAPE
EXECUTION OF MODIFICATION
CHECK MODE
INPUT OF DIGITAL VOLTMETER READINGS
CALL UP PRESTORED ANALYSED PROBLEM STRUCTURE FROM TAPE
COMPUTATION OF NUMERICAL VALUES
OUT OF TOLERANCE ?
YES
NO
LISTING OF ERRORS
OK
END OF CHECK
ANALOGUE COMPUTER SET - UP

Fig. 2—APACHE program organization.

The average operator will immediately produce programs giving correct results, while the expert will find that full scope is allowed for more subtile use of the machine, since any circuit which can be patched with standard elements can be written in APACHE code. On the other hand, all users will find that the tedious and mechanical elements of programing, namely calculation of coefficient settings and static check, and the onerous fault-finding procedures have been entirely eliminated, leading to a much more efficient use of trained personnel.

It is intended that the manner of working in the production stage shall be more systematic than hitherto and that as far as possible advantage be taken of the possibility of producing production tapes automatically. Thus, it is intended to increase the loading of the analog machines substantially because of

1) The very large reduction in "debugging" time,
2) The greatly increased possibility of "timesharing" between several problems, due to the rapid checking procedure which gives much cheaper thinking time and, thus, the chance to prepare for faster and more systematic production.

It is to be noted that during operation, changes in patching and coefficients may be made manually with perfect freedom, and it is only necessary to keep a careful note of them for perforation and addition to the program at any convenient time, while it is also possible to introduce modifications into the stored program which are treated in a few seconds of digital-machine time and which put out complete or partial listings and miniature setting and checking tapes.

Status of the Program and Future Prospectus

A limited version called APACHE 1 has been in operation since March, 1961, at Ispra, and it is on the basis of this restricted version that APACHE 2 has been designed. The new version is scheduled to be in operation by the time this article is printed.[3] Although several references have already been found, and in many places in the United States and in England work has been attempted which covers a part of the automatic analog programing problem, nothing has so far come to hand of a similar scope and magnitude. Thus, diffusion is sought particularly for the language employed, in the hope that it may be found a suitable standard for future work in this field.

In this connection it has been a constant aim of the study to create a programing and checking system, which while being completely self-contained is designed to be part of a fully integrated system of analog and

[3] APACHE II came into operation at Ispra in July, 1962, and copies have been distributed to several other computation centers.

digital computers, and it is claimed that APACHE is already prepared to accept the additions necessary for the possibilities which follow. The first of these is the remote-controlled patching system which is already under active study, and which would bring almost the whole problem set-up under the direct control of the digital machine.

If desired, the communication could at this stage be made purely electrical, eliminating the use of punched cards. With the addition of remote-controlled function and delay generators it completes what is known as the static phase of the integration of the two types of computation which would necessitate a physical reorganization of the next generation of analog machines, leading to much greater flexibility in assembling suitable machines to solve a given problem.

A corollary to this must surely be the replacement of the present wide variety of multiplier types by a standard unit with high accuracy and effectively unlimited frequency response, greatly simplifying programing. Another highly important aspect is the automation of maintenance procedures which will change completely the problem of machine realiability when it is realized.

This is not to forget the dynamic link, using the analog-digital and digital-analog converters which are already commercially available, enabling the two types of machine to work simultaneously on one large problem. There can be no doubt that this technique which has already been proven powerful for some classes of problems can be greatly developed once the setup of the analog machine is brought into line with that of the digital computer.

Appendix

This appendix is intended primarily to summarize the language above and the reader should consult the publication described[2] for a definitive statement on the input and output language. These notes should make the above example quite clear however.

Number: There are two forms permitted.

1) As in normal use including decimal point in any position.
2) As above with an exponent of 10 added after a capital E—*e.g.* $12.51E-2=12.51\times10^{-2}=0.1251$.

For the output format a more specific rule is used by the machine, the exponent being employed if, and only if, the number is greater than 9999 or less than 0.01.

Name: A name consists of up to 5 alphabetical or numerical characters, of which the first must be a letter, and can refer to either a parameter or a variable.

Arithmetic Symbols: Keep their usual meanings with certain machine conventions added as in FORTRAN. This applies also to brackets. The asterisk (*) is the sign of multiplication and the slash (/) of division.

Orders: These are given in capitals below, and at the input may be taken in any order. They are cancelled by the next order given.

PARAMETERS is followed by a list of constants and their values. A parameter can be composed of more than one constant and its value can be a combination of other parameters provided only that the complete list be self-consistent. Parameters introduced directly into the equations in numerical form need not be listed.

VARIABLES precedes a list of names, to which are attached the following columns of information, *all in physical units:*

1) Initial value, where appropriate. Redundant values will be ignored by the program, which replaces them by calculated values,
2) Maximum value, from which a scale factor will be calculated and rounded off for easy manipulation,
3) Minimum value, if it is desired that the variable in question be expressed as a perturbation about its mean,
4) Comments such as:
   LIMITS indicating the use of a feedback limiter,
   LIMITS B for a bridge limiter,
   TEST causing the initial condition to be patched to the machine test reference,
   EXACT inhibiting the round off of the scale factor so as to make the maximum value correspond to precisely 100 volts.

EQUATIONS precedes the *physical* equations, the rules of writing being conveniently explained for the following classes:

1) Linear and nonlinear ordinary differential equations of any order. The highest derivative of the variable for which the equation is solved is placed on the left, the operator d^n/dt^n being replaced by the operator DER $n($ $)$. (n being any number from 0 to 9.)

   If DER $n(x)$ is included in the list of variables, the equation will be treated as algebraic and DER $n(x)$ calculated explicitly. Otherwise, the right hand side will be integrated to yield DER $(n-1)(x)$. Multiplications and divisions are admissible on the right hand side at the input, these being automatically converted into complementary equations by the program, but other types of nonlinearity must be replaced by dummy variables for which equations are given elsewhere. Evidently, the type of multiplier and sometimes the number of quadrants must also be given for multiplication and division, if more than one type is available.
2) Algebraic equations are covered by the same rules except that summation is implied by the absence of the operator. The variable on the left must be in the variable list and no initial condition is required for it.
3) Implicit equations are resolved by the operator ZERO, the variable for which the equation is to be solved being written on the left. Thus the equa-

tion for a square root, $y^2-x=0$ is written ZERO $(y)=y^2-x$.

The operator ZERO corresponds to a high-gain amplifier in the analog circuit.

4) Arbitrary functions are written as equations using special operators which give all necessary information about the analog set-up. Thus $y=f(x)$ is written DFG $20(y)=$ FUNCTION (x) in the case of a diode function generator of 20 segments or $y=zf(x)$
$PPD\ (Y)=(\text{FUNCTION}\ (X))*Z$
in the case of a servomultiplier with a padded pot. A fixed function generator is represented by the operator FFG.

5) Switching operations use the operators COMPARE for comparators and SWITCH for manual switches and the reader will have no difficulty in understanding the example, provided that he remembers that the output of an element follows the operator on the left while the input(s) come on the right, separated by arithmetical symbols or commas.

6) Resolutions of coordinates are carried out by resolvers and rate resolvers which are represented by the operators
RES, RECTANGULAR
RES, POLAR
RRV, RECTANGULAR
RRV, POLAR
which are self-explanatory.

Other orders are provided to enable the operator to control the problem set up and need be used only in certain cases. These are:

IMPOSE which allocates a given variable or parameter to an address chosen by the programer.

OMIT which prevents the allocation of the addresses which follow.

SUPPRESS effaces a given list of statements.

INSERT n adds the statements which follow after statement *n*.

COMMENTS precedes a statement which will be reproduced but will not affect the program.

VARIPLOTTER and RECORDER assign to these elements a given list of variables (for automatic patching).

REFERENCE is used for those machines which have a scale other than ±100 volts.

CONSOLE SELECT allows the distribution of the program among several consoles.

BETA imposes the time scale.

These last two will be automatic if no specification is made. In the case of BETA a time scale is chosen which keeps the largest gain slightly below 10.

In addition to these orders, which allow the programing stage to be completed in the classical manner, some others have been added, giving new possibilities.

MODIFY recalls a program for the incorporation of changes.

SUBROUTINE allows the insertion of a preprogramed block corresponding to a "black box" at the set-up level, the arguments of the subroutine including the scale factors of input and output variables.

FUNCTION has a similar purpose except that a function has one output and a subroutine several.

RETURN terminates a function or a subroutine.

END signals the completion of the problem.

CHECK prepares the entry of a readout from the analog machine for analysis by the digital program.

COMPUTE prepares the necessary data to carry out a program of production in whatever form the installation can assimilate it.

Acknowledgment

The authors wish to acknowledge the collaboration of G. P. Del Bigio who wrote most of the APACHE code and contributed many important ideas to the project. We wish to thank also MM. P. Braffort, A. Gazzano and A. Riotte for their good advice and for the support they gave to this work.

Bibliography

[1] L. Ohlinger, "ANATRAN—Fist step in breeding DIGNALOG," *Proc. the 1960 Western Joint Computer Conf.*, pp. 315–328.
[2] T. Witzel and J. L. Wilson, "Digital computer programs analog solution," *Control Engrg.*, vol. 8, pp. 135–138; June, 1961.
[3] J. Diebold, *et al.*, "The hybrid computer," *Automatic Data Processing*, vol. 3, pp. 20–23; July, 1961.
[4] G. P. Del Bigio, "RAPPORT APACHE 1, EUR-C-IS/441/61 f. (Unpublished.)
[5] C. Green, A. Debroux, G. P. Del Bigio, and H. d'Hoop, "Le code APACHE, destinè á la programmation d'un problême analogique au moyen d'un calculateur digital," *Proc. 3rd Internatl. Conf. on Analog Computing*, Opatija, Yugoslavia, Presses Acadêmiques Europêennes, Brussels, Belgium, Commun. No. 107; September, 1961.
[6] P. Camion, "Nombre minimum d'inverseurs dans un schêma analogique," Cetis, Euratom, Ispra, Italy. (Unpublished.)