Three-In-One Curve Fits: Hyperbolic, Exponential, Power Functions

7.0.019

DISCLAIMER

Although each program has been tested by its contributor, no warranty, express or implied, is made by the contributor or any User's Group, as to the accuracy and functioning of the program and related program material, nor shall the fact of distribution constitute any such warranty, and no responsibility is assumed by the contributor or any User's Group, in connection therewith.

Modifications or revisions to this program, as they occur, will be announced in the appropriate Catalog of Programs for IBM Data Processing Systems. When such an announcement occurs, users should order a complete new program from the Program Information Department.

COMMON USERS GROUP PROGRAM REVIEW AND EVALUATION

(fill out in typewriter, ink or pencil)

Program No.________________ Date________________

Program Name:____

1. Does the abstract adequately describe what the program is and what it does? Yes____ No____
   Comment____

2. Does the program do what the abstract says? Yes____ No____
   Comment____

3. Is the description clear, understandable, and adequate? Yes____ No____
   Comment____

4. Are the Operating Instructions understandable and in sufficient detail? Yes____ No____
   Comment____
   Are the Sense Switch options adequately described (if applicable)? Yes____ No____
   Are the mnemonic labels identified or sufficiently understandable? Yes____ No____
   Comment____

5. Does the source program compile satisfactorily (if applicable)? Yes____ No____
   Comment____

6. Does the object program run satisfactorily? Yes____ No____
   Comment____

7. Number of test cases run_____. Are any restrictions as to data, size, range, etc. covered adequately in description? Yes____ No____
   Comment____

8. Does the Program meet the minimal standards of COMMON? Yes____ No____
   Comment____

9. Were all necessary parts of the program received? Yes____ No____
   Comment____

10. Please list on the back any suggestions to improve the usefulness of the program. These will be passed onto the author for his consideration.

Please return to:

IBM Corporation
Program Information Department
40 Saw Mill River Road
Hawthorne, New York 10532
Attn: PREP FORM COORDINATOR

Your Name ____________________
Company ____________________
Address ____________________
____
Users Group Code ____________________

THIS REVIEW FORM IS PART OF THE COMMON ORGANIZATION'S PROGRAM REVIEW AND EVALUATION PROCEDURE. NONMEMBERS ARE CORDIALLY INVITED TO PARTICIPATE IN THIS EVALUATION.

2/2/67

PROGRAM DESCRIPTION

This program is as nearly as possible a general purpose program for the frequently encountered curves $y = b+c/(x-a)$, $y = ab^x$, and $y = ax^b$. It is really three subprograms, and manual branching via 1620 console is necessary to move from one subprogram to another. All output is via the console typewriter. All input is via the 1622 card reader or console typewriter (OPTION 5 only). For each subprogram the options listed below are available to the user.

OPTION 1

Fit by Least-Squares Method a curve of desired form to a set of empirical data. When Op 1 is chosen, batch processing is possible. Normally batch processing would be used when experience dictated form of curve to be used for several sets of data.

OPTION 2, "2a"

Op 1 plus standard error and plot back of as many points as desired. Interpolated values furnished also. Batch processing is not possible, because two passes through the card reader must be made with the same set of observed data. A third pass is used for interpolation.

OPTION 3

No interpolation; otherwise same as Op 2.

OPTION 4

Op 1 plus plot back of a few select points and/or interpolation. No standard error computation. This is a two-pass option with a three-pass function, and careful reading of operating instructions is recommended.

OPTION 5

Function evaluation by keying-in parameters at console typewriter. Use of this feature permits evaluation of any function of specified form, provided arithmetic and function values remain within bounds of FORTRAN floating subroutines. Feature especially useful when interpolation is "afterthought".

METHOD OF COMPUTATION

LEAST SQUARES FIT: (HYPERBOLA)

$y = b+c/(x-a) \Rightarrow (y-b)(x-a) = c \Rightarrow xy-bx-ay+ab = c \Rightarrow xy = (c-ab)+bx+ay$. Least squares method is used to fit last form to set of n observed data (x_o,y_o). Three normal equations in the three unknowns, (c-ab), b, and a, result. We solve these simultaneously:

$$n(c-ab)+b\sum_{o=1}^{n} x_o + a\sum_{o=1}^{n} y_o = \sum_{o=1}^{n} x_o y_o \qquad (1)$$

$$(c-ab)\sum x_o + b\sum x_o^2 + a\sum x_o y_o = \sum x_o^2 y_o \qquad (2)$$

$$(c-ab)\sum y_o + b\sum x_o y_o + a\sum y_o^2 = \sum x_o y_o^2 \qquad (3)$$

Third equation was solved for (c-ab), and result was substituted in (1) and (2) from which a was then eliminated. Resulting expression for b required two FORTRAN statements. Value a obtained by formula resulting from substitution of b in modified second equation. Value (c-ab) obtained by substitution of a and b into formula given by initial solution of (3) for (c-ab). Subsequent solution for c is immediate.

LEAST SQUARES FIT: (EXPONENTIAL)

$y = ab^x \Rightarrow \ln y = \ln a + (\ln b)x$. Let $\ln y = y'$, $\ln a = A$, $\ln b = B$. Hence $y' = A + Bx$. To fit line of this form to a set of n observed data (x_o,y_o), we perform the transformation $y_o' = \ln y_o$. Then we solve simultaneously for A and B the normal equations:

$$\sum_{o=1}^{n} y_o' = n\cdot A + B\sum_{o=1}^{n} x_o \qquad (4)$$

and $$\sum x_o y_o' = A\sum x_o + B\sum (x_o)^2 \qquad (5)$$

for B by Gaussian elimination and for A by substitution in (4). Exponentiation of A and B gives a and b for output.

LEAST SQUARES FIT: (POWER FUNCTION)

$y = ax^b \Rightarrow \ln y = \ln a + b(\ln x)$. Let $\ln y = y'$, $\ln a = A$, $\ln x = x'$. Hence $y' = A + bx'$. To fit a line of this form to a set of observed data (x_o,y_o), we perform the transformations $x_o' = \ln x_o$ and $y_o' = \ln y_o$. Then we solve simultaneously for A and b the normal equations:

$$\sum_{o=1}^{n} y_o' = n\cdot A + b\sum_{o=1}^{n} x_o' \qquad (6)$$

and $$\sum x_o' y_o' = A\sum x_o' + b\sum (x_o')^2 \qquad (7)$$

for b by Gaussian elimination and for A by substitution in (6). Exponentiation of A gives a for output.

COMPUTATION FOR STANDARD ERROR, PLOT BACK AND FUNCTION EVALUATION:

Once parameters have been computed, machine can compute directly y_c, the plot back value. The deviation, $y_d = y_o - y_c$, is readily available for typeout and for squaring and summing in computing the standard error.

TABLE OF CONTENTS

(This write-up contains 31 pages.)

Three-in-One Curve Fits
Hyperbolic, Exponential, Power Functions

by

Wade A. Norton
Southern Services, Inc.
600 North 18th Street
Birmingham 2, Alabama

DISCLAIMER

25 July 1962
7.0

2B

Interpolation and function evaluation are also done directly by FORTRAN using dummy y_0 values. Dummies do not appear as output.

ERROR ANALYSIS:

Interrelatedness of summations used for least squares fit makes general statements about error analysis difficult. Individual problem approach seems best regarding reliability of computed parameters.

PROGRAM CREDITS:

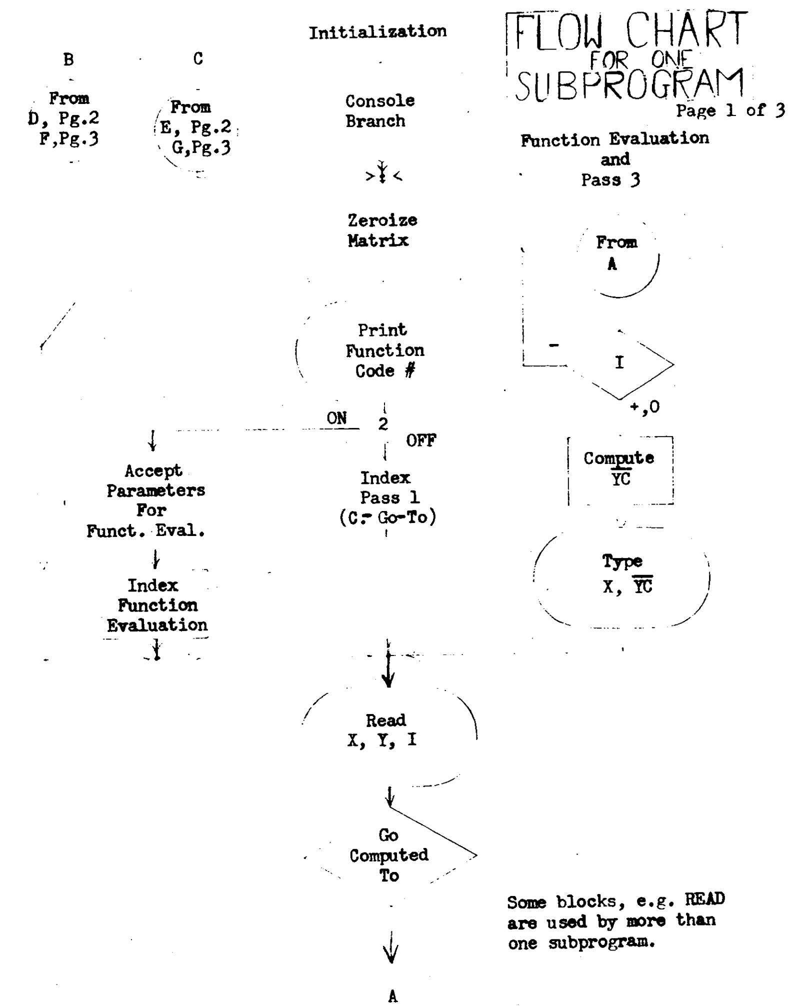
Southern Services, Inc., Birmingham, Alabama, is contributor of this program.

Inquiries and comments should be addressed to E. J. Orth, Jr., director, Computing Center, at the above address or telephone: Birmingham FA3-5341, Ext. 2339. He contributed much to this program.

We wish to thank Dr. R. E. Wheeler, head of the mathematics department, Howard College, Birmingham, for his constructive comments and Mr. Jim Moore, IBM, Birmingham, for permission to include his FORTRAN Compressor.

We wish also to thank our programming colleagues, Mrs. Mary T. Shannon and John L. Redmond, for their assistance. The final manuscript is the careful work of typists, Mrs. Tallulah S. Armistead and Miss Betty Ross Armstrong. Our machine room supervisor, Jim Ewing, says he understands the Operating Instructions, so hopefully we submit the program.

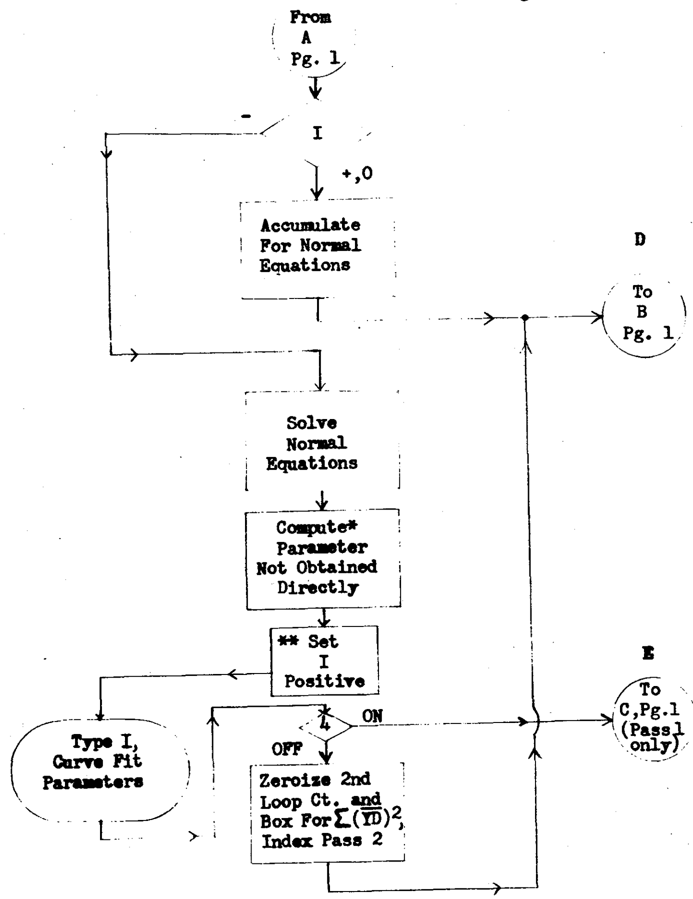
Wade A. Norton, author
Birmingham, Alabama
25 July 1962



Some blocks, e.g. READ are used by more than one subprogram.

PASS 1

FLOW CHART
Page 2 of 3

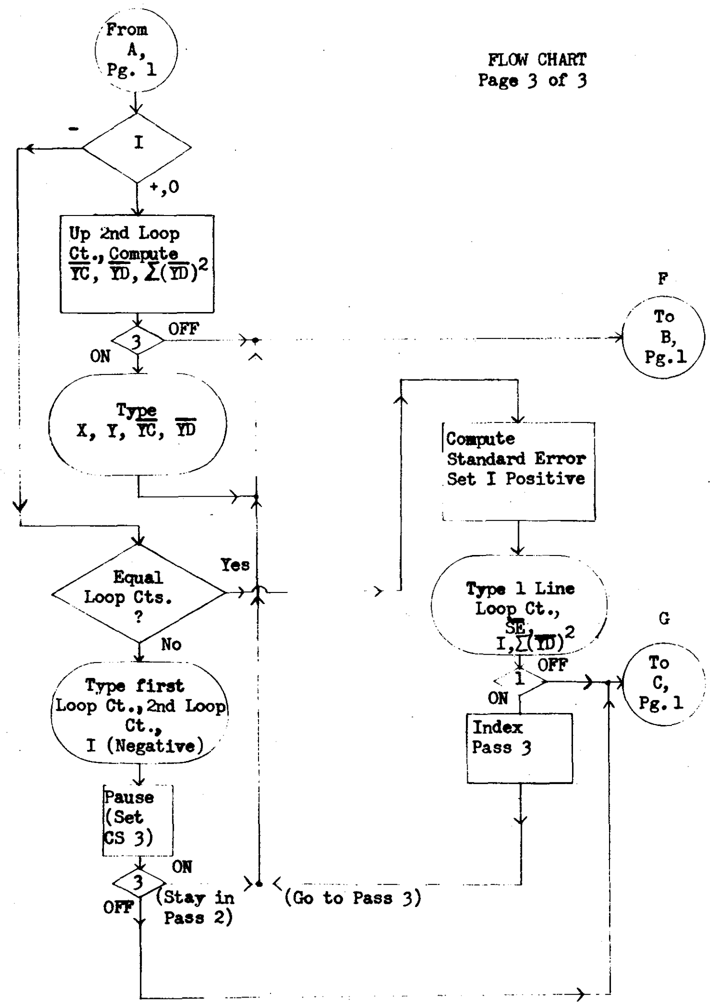


*For Power Function
This is an internal
x'fer of parameter only.

**For Exponential and Power Functions
order of this and preceding block
are reversed. This is hyperbolic order.

Pass 2

FLOW CHART
Page 3 of 3



INPUT

CARD INPUT:

All card input is one point (observation) per card in the form:

x_o, y_o, I

where I is fixed point identification number processed by 1620 and must appear entirely in columns numbered $\leq$ 72. Form of x_o and y_o is floating point. End-of-file card contains dummy (x_o, y_o) and negative value for I. Dummy y_o values are used for function evaluation and interpolation, thus making all card input compatible.

TYPED INPUT:

Under Option 5, two (or three) floating point values are keyed-in at console typewriter:

a RS
b RS
c RS (c not accepted unless hyperbola).

OUTPUT (ALL TYPED)

FUNCTION CODE: Fixed point digit 1, 2, or 3 on line by itself indicates subprogram in use: Hyperbolic, exponential, or power, respectively.

CURVE FIT (Pass 1):

Line 1: I, a, b, c (c only for hyperbola)
Line 2: a, (ln b) (Line 2 only for exponential)

where I is absolute value of fixed point identification number appearing in end-of-file card, and a and b (and if c) are computed parameters for curve. For exponential fit, value a also represents the y-intercept (for x = 0) of the linear (semi-log) form, and (ln b) the semi-log slope.

PLOT BACK (Pass 2):

x_o, y_o, y_c, y_d

where x_o and y_o are the observed values, y_c is the computed y for fitted curve, and $y_d = y_o - y_c$.

STANDARD ERROR (Pass 2):

n, $\overline{SE}$, I, $\sum (y_d)^2$

where $\overline{SE}$ is floating point computation of standard error and others previously defined.

INTERPOLATION (Pass 3) AND FUNCTION EVALUATION:

x_o, y_c

where variables indicated are same as described above. "Observed" y_o is

not printed, as it is a dummy used only to make all input compatible.

PROGRAMMED ERROR MESSAGE (Pass 2, identified by all fixed point output with last negative):

N, M, -I

where N is first pass count, M is second pass count, and I is negative, just as read from end-of-file card. A meaningful standard error can be computed only on data used to compute parameters. Program utilizes three-pass system: (1) compute parameters, (2) compute standard error and plot back, and (3) interpolate. By using three passes, we place no inherent limitations on n, number points used in fit. The 1620 cannot be sure that same data are processed on Passes 1 and 2 but can be sure that same number of cards N, M are processed during respective passes before computing standard error. When same number of points are not processed each pass, standard error is not computed, and first and second pass counts respectively are typed. In one sense this is NOT NECESSARILY an error message, as user may not intend to use same number of points each pass. (See Option 4, page 16). Number of interpolated values is independent of fit, so no count of Pass 3 points is kept.

FORTRAN ERROR MESSAGES: On Pass 2, a zero divisor will result in FORTRAN message ERROR E7 while computing the standard error when n = 3 for the hyperbola or n = 2 for the exponential or power functions. This does not affect the plot back, but the standard error itself should be ignored. The program is subject to other FORTRAN error messages, but none should occur in normal operation. Should the messages ERROR Ex or ERROR Fx (where x is a digit) appear on the typewriter, consult the FORTRAN write-up. Use of fewer points than specified above results in insufficient data to define uniquely the curve in question, and a variety of error messages of this sort are likely. Under such conditions the output should be ignored, but the program remains intact.

SAMPLE PROBLEMS

SAMPLE PROBLEM #1: HYPERBOLA

The 1620 was given the 12 points listed in the plot back for this problem. These points are known to lie on both continuous curves comprising the hyperbola $(x+7)(y+2) = 30$. Results are obvious from glance at plot back and interpolation.

SAMPLE PROBLEM #2: HYPERBOLA

The 1620 was given the 8 points listed last in the plot back for SAMPLE PROBLEM #3. These points are known to lie on both continuous curves comprising the hyperbola $xy = 12$. Neither plot back option nor interpolation option was used. Success is obvious.

SAMPLE PROBLEM #3: HYPERBOLA

The 1620 was given the 20 points comprising the data for SAMPLE PROBLEM #1 and SAMPLE PROBLEM #2. Computed values were checked by desk calculator. Standard error is high, because points do not cluster about a single hyperbola.

SAMPLE PROBLEM #4: HYPERBOLA

The 1620 was given the parameters $a = 2$, $b = -3$, $c = 360$. Function evaluation for $x_o = 4, 6$ gave $y_c = 177, 87$ respectively.

SAMPLE PROBLEM #5: EXPONENTIAL

The 1620 was given the 22 points listed in the plot back of this problem. These are really two sets of points: eleven points on $y_2 = 2e^x$ and eleven points on $y_3 = (e^x)/2$. Observation of output shows program fitted $y_1 = e^x$ to accuracy of input. Standard error is high, because test data fail to cluster about single curve.

SAMPLE PROBLEM #6: EXPONENTIAL

The 1620 was asked to evaluate $y = 3e^x$ for $x_o = 0, 1, 2, \ldots, 10$; ie., $a = 3$ and $b = 2.7183$ were keyed-in at console typewriter. Results compare favorably with desk methods based on identical "input".

SAMPLE PROBLEM #7: POWER FUNCTION

The 1620 was given the 10 points listed in the plot back for this problem. Five of these points lie on ex^2 and five on $ex^{\frac{1}{2}}$. Observation of output shows program fitted $ex^{1.25}$ to accuracy of input. Standard error is high, because data fail to cluster about a single curve.

SAMPLE PROBLEM #8: POWER FUNCTION

The 1620 was asked to evaluate $y = 2x^3$ for x_o values listed in typeout; ie., $a = 2$, $b = 3$. Output is slightly off because exponent is floating point, necessitating use of subroutine by FORTRAN.

SAMPLE PROBLEMS

LISTING

Sample Problem #1, CS $\begin{cases}10\\10\end{cases}$

1			
4444	-7.0000000	-2.0000000	30.000000
-5.0000000	13.000000	13.000000	.00000000
-2.0000000	4.0000000	4.0000000	.00000000
-1.0000000	3.0000000	3.0000000	.00000000
3.0000000	1.0000000	1.0000000	.00000000
8.0000000	.00000000	.00000000	.00000000
23.000000	-1.0000000	-1.0000000	.00000000
-9.0000000	-17.000000	-17.000000	.00000000
-12.000000	-8.0000000	-8.0000000	.00000000
-13.000000	-7.0000000	-7.0000000	.00000000
-17.000000	-5.0000000	-5.0000000	.00000000
-37.000000	-3.0000000	-3.0000000	.00000000
53.000000	-1.5000000	-1.5000000	.00000000
12	.00000000	4444	.00000000
4.0000000	.72727270		
6.0000000	.30769230		
1			

Sample Problem #2, CS $\begin{cases}00\\00\end{cases}$

1			
3333	-.00000000	.00000000	12.000000
8	.00000000	3333	.00000000
1			

Sample Problem #3, CS 00/10

1

8888	-5.2202116	-1.8693454	37.272553
-5.0000000	13.000000	167.38856	-154.38856
-2.0000000	4.0000000	9.7052210	-5.7052210
-1.0000000	3.0000000	6.9625702	-3.9625702
3.0000000	1.0000000	2.6649117	-1.6649117
8.0000000	.00000000	.95001600	-.95001600
23.000000	-1.0000000	-.54857030	-.45142970
-9.0000000	-17.000000	-11.730361	-5.2696390
-12.000000	-8.0000000	-7.3669431	-.63305690
-13.000000	-7.0000000	-6.6602919	-.33970810
-17.000000	-5.0000000	-5.0334558	3.3455800E-02
-37.000000	-3.0000000	-3.0421836	4.2183600E-02
53.000000	-1.5000000	-1.2291459	-.27085410
4.0000000	3.0000000	2.1731380	.82686200
6.0000000	2.0000000	1.4525665	.54743350
3.0000000	4.0000000	2.6649117	1.3350883
2.0000000	6.0000000	3.2929067	2.7070933
12.000000	1.0000000	.29512010	.70487990
1.0000000	12.000000	4.1228226	7.8771780
24.000000	.50000000	-.59377110	1.0937711
-4.0000000	-3.0000000	28.676629	-31.676629
20	38.343302	8888	24993.553

1

Sample Problem #4, CS 01/00

1

+2.RS
-3.RS
360.RS

4.0000000	177.00000
6.0000000	87.000000

1

Sample Problem #5, CS 10/10

2

1111	1.0000108	2.7182750	
1.0000108	.99999752		
.00000000	.50000000	1.0000108	-.50001080
1.0000000	1.3592000	2.7183043	-1.3591043
2.0000000	14.778200	7.3890988	7.3891020
3.0000000	10.043000	20.085601	-10.042601
4.0000000	109.19600	54.598188	54.597820
5.0000000	74.200000	148.41288	-74.212880
6.0000000	806.86000	403.42705	403.43295
7.0000000	548.30000	1096.6256	-548.32560
8.0000000	5962.0000	2980.9299	2981.0701
9.0000000	4051.6000	8102.9873	-4051.3873
10.000000	44052.000	22026.148	22025.852
.00000000	2.0000000	1.0000108	.99998920
1.0000000	5.4366000	2.7183043	2.7182957
2.0000000	3.6946000	7.3890988	-3.6944988
3.0000000	40.172000	20.085601	20.086399
4.0000000	27.299000	54.598188	-27.299188
5.0000000	296.82000	148.41288	148.40712
6.0000000	201.72000	403.42705	-201.70705
7.0000000	2193.2000	1096.6256	1096.5744
8.0000000	1490.5000	2980.9299	-1490.4299
9.0000000	16206.200	8102.9873	8103.2130
10.000000	11013.000	22026.148	-11013.148
22	5921.7808	1111	7.0134993E+08
1.5000000	4.4817204		
9.5000000	13359.551		

2

Sample Problem #6, CS 01/00

2

3.RS
2.7183RS

.00000000	3.0000000
1.0000000	8.1548991
2.0000000	22.167460
3.0000000	60.257802
4.0000000	163.79877
5.0000000	445.25415
6.0000000	1210.3343
7.0000000	3290.0514
8.0000000	8943.3459
9.0000000	24310.695
10.000000	66083.757

2

Sample Problem #7, CS $\left\{ \begin{matrix} 10 \\ 10 \end{matrix} \right.$

3			
2222	2.7182994	1.2500001	
1.0000000	2.7183000	2.7182994	6.0000000E-07
4.0000000	5.4366000	15.377024	-9.9404240
9.0000000	220.18230	42.374096	177.80821
16.000000	10.873200	86.985603	-76.112403
25.000000	1698.9375	151.95760	1546.9799
1.0000000	2.7183000	2.7182994	6.0000000E-07
4.0000000	43.492800	15.377024	28.115776
9.0000000	8.1549000	42.374096	-34.219196
16.000000	695.88480	86.985603	608.89920
25.000000	13.591500	151.95760	-138.36610
10	593.98197	2222	2822519.0
1.5000000	4.5124393		
9.5000000	45.336904		
3			

Sample Problem #8, CS $\left\{ \begin{matrix} 01 \\ 00 \end{matrix} \right.$

3	
2.05	
3.05	
.00000000	.00000000
1.0000000	2.0000000
1.5000000	6.7499996
2.0000000	15.999999
3.0000000	53.999984
4.0000000	127.99997
5.0000000	249.99998
3	

COMPILATION INSTRUCTIONS

Since the program is supplied in source form, it will be necessary for the user to compile the program himself. Note that input/output is designed for pre-Format FORTRAN. The source deck is numbered serially in columns 78-80. To assist in the location of cards in the source deck, the first, last, and every tenth statement in the program listing have been numbered at the right margin. This number corresponds to the serial number of the card in which the statement appears.

OPERATING INSTRUCTIONS

Clear Core.

Set Check Switches: Overflow to Program,
I/O to Stop,
Parity to Stop.

Set typewriter tab stops 17 spaces apart for FORTRAN output.

Load Program.

Set CS for desired option as explained below.

Branch to subprogram: 07500 for hyperbola,*
10312 for exponential,
12596 for power.

The appropriate function code will be printed.

When the Reader-No-Feed light comes on, read in data with end-of-file card. Or, according to option chosen, if typewriter enables, enter function parameters (see page 7).

After subprogram is executed, another subprogram may be branched to.

Console Switch Settings**

Option 1: Compute Equation Parameters Only: 0001

Option 1 allows batch processing. Each set of observations must have an end-of-file card; and after each set is processed the function code is printed. When desired set(s) of data has/have been processed, the user may reset the console switches (if desired) and branch to another subprogram.

Option 2: Compute Equation Parameters, Plot Back, Standard Error, and Interpolation: 1010

Only one data file may be processed at a time, since two passes through the Reader must be made. On first pass, one or two lines of curve fit output will appear. A Reader-No-Feed light also signals the beginning of Pass 2, where plot back and standard error are computed. After data file is processed by Pass 2, interpolation points should be entered in the Reader. Interpolation points consist of x_o, dummy y_o, and I. Dummy y_o values and I are not printed. Interpolation cards must have their own end-of-file card. After interpolation data are read, the function code is printed. This is the time to branch to another subprogram (after setting switches as desired).

If the card counts for Passes 1 and 2 are not equal, an error message will print after the end of Pass 2 in the form of 3 numbers: number of cards in Pass 1, number of cards in Pass 2, and -I. To correct, see note 3, Console Program Switches, page 17.

* See Comments, page 18.
** See Note 9, page 26.

Option 3: All Features of Option 2 except Interpolation: 0010

After Pass 2 the function code is printed. This is time to branch to another subprogram (after setting switches as desired).

Option 4: Compute Parameters, Some Plot Back, Some Interpolation: 0010

Option 4 is a variation on Option 3 and uses the same switch settings. Procedure is the same except as noted below. Option 4 will be applied under these conditions:
(a) standard error and sum of squared deviations are not desired,
(b) only partial plot back desired,
(c) interpolation is or is not desired.
The data file for Pass 2 consists of all points to be plotted whether plot back or interpolation. Pass 2 data file must have an end-of-file card. On interpolation, y_o and y_d of plot back will appear and should be disregarded. Also ignore the last line of print before the typing of the function code: it will be either an erroneous standard error or an incorrect error message. Option 4 ends after typing of the function code. When the halt (48 in op register) occurs because of false error message, turning CS3 OFF and hitting start causes typing of the function code.

Option 5: Function Evaluation: 0100

This option is especially valuable when interpolation or plot back is "after-thought". When 37 op code appears in console op register, key-in function parameters. When Reader-No-Feed light appears, load data file which consists of data cards (containing x_o, dummy y_o, I) and end-of-file card (where both x_o and y_o are dummy values and I is negative). After the evaluation is completed, the function code is typed.

CONSOLE PROGRAM SWITCHES (DETAILED DISCUSSION)

All comments below refer to use of a single subprogram.

1. CS2. In Pass 1 of Options 1 through 4, when Reader-No-Feed light appears on console, CS2 has been interrogated. It may be turned ON now or at any time before last pass is completed, if next run is to be function evaluation. When using Option 5, best place to turn CS2 OFF (if next run is fit) is at Reader-No-Feed before type-out. After completing any option, machine types function code and goes to one of two program points. With CS2 ON, 37 appears in op register to indicate function evaluation block has been entered. With CS2 OFF Reader-No-Feed light appears to indicate Pass 1 block has been entered.

2. CS3. Plot back can be turned OFF or ON while in progress. An "Option 2a" involving three passes will yield parameters, limited plot back, standard error, and interpolation. Settings and steps are as for Option 2 with following modification: (a) Arrange observed data file, so points for which plot back desired are at front of file on Pass 2, (b) Turn CS3 OFF while last point of desired plot back is being typed, (c) Use Pass 3 for interpolation. Such "Option 2a" could replace Option 4. Op 4 has advantages of less machine time and requiring no change of CS settings during run. Op 2a has advantages

of consistency with Op 2, valid standard error furnished, and no dummy y_0 typeout for interpolation.

3. CS3. After "ERROR MESSAGE" (N, M, -I) machine halts. If true error has occurred, turning CS3 ON permits repetition of Pass 2: To repeat Pass 2, clear Reader, locate error, load Reader-Hopper with correct data file and hit Start. Remember CS3 controls plot back, so when repeating Pass 2, change CS3 setting, if necessary, before hitting Reader-Start. With CS3 OFF program returns to type function code and interrogate CS2.

4. CS4. CS4 is interrogated at the end of Pass 1. If CS4 is ON, CS2 is interrogated as stated above. If CS4 is OFF, the program continues to Pass 2.

5. CS1. CS1 is interrogated at the end of Pass 2 when no error condition detected. If CS1 is ON, the program continues to Pass 3 for interpolation. If CS1 is OFF, CS2 is interrogated as stated above.


```
C   THIS PROGRAM FITS BY METHOD OF LEAST SQUARES CURVES OF FORM
C     Y=B+C/(X-A) (AN HYPERBOLA), Y=A(B**X) (AN EXPONENTIAL), AND
C     Y=A(X**B).  LEAST SQUARES METHOD IS APPLIED DIRECTLY IN CASE OF
C     HYPERBOLA AND TRANSFORMATION TO LINEAR FORM (SEMI-LOG OR LOG-LOG
C     RESPECTIVELY) IS MADE IN OTHER TWO CASES.  MANUAL BRANCHING VIA
C     1620 CONSOLE IS NECESSARY TO MOVE FROM ONE TYPE FIT TO ANOTHER.
C     (FOR BRANCHING DETAILS, SEE OPERATING INSTRUCTIONS.)  FIRST PART OF
C     PROGRAM IS HYPERBOLIC FIT WHICH FOLLOWS.  BRANCHING IS NOT NECES-
C     SARY TO ENTER IT INITIALLY, BUT IS OTHERWISE.
      DIMENSIONS(2,4)
   03 DO5K=1,2
      DO5J=1,4
   05 S(K,J)=0.
      PRINT,
      PRINT,1
      PRINT,
      MM=3
      IF(SENSE SWITCH2)1,12
    1 ACCEPT,S(2,4),B,C
      L=2
      GO TO 6
   07 IF(I)3,4,4
    4 S(1,2)=B+(C/(X-S(2,4)))
      GO TO 1429
   12 L=1
   06 READ,X,Y,I
      GO TO (2,7,51,7,1007,1032,1129,1129,2007,2032,2129,2129),L
   02 IF(I)10,8,8
   08 S(1,3)=S(1,3)+1.
      S(1,1)=S(1,1)+X
      S(2,1)=S(2,1)+X**2
      S(1,2)=S(1,2)+Y
      S(2,2)=S(2,2)+X*Y
      S(2,3)=S(2,3)+Y*Y
      S(2,4)=S(2,4)+X*X*Y
      S(1,4)=S(1,4)+X*Y*Y
      GO TO 6
   10 B22=S(1,2)*S(2,2)-S(1,1)*S(2,3)
      B21=S(2,1)*S(1,2)-S(1,1)*S(2,2)
      B23=S(1,2)*S(2,4)-S(1,1)*S(1,4)
  B=B22*(S(2,2)*S(1,2)-S(1,3)*S(1,4))-B23*(S(1,2)**2-S(1,3)*S(2,3))
  B=B/(B22*(S(1,1)*S(1,2)-S(1,3)*S(2,2))-B21*(S(1,2)**2-S(1,3)*S(2,3)))
  S(2,4)=(B23-B21*B)/B22
  C=(S(1,4)-S(2,2)*B-S(2,3)*S(2,4))/S(1,2)
      C=C+S(2,4)*B
      I=-I
      PRINT,I,S(2,4),B,C
      PRINT,
      N=S(1,3)
      IF(SENSE SWITCH4)3,54
   54 L=3
      GO TO 1131
   51 IF(I)49,50,50
   50 M=M+1
      S(1,2)=B+(C/(X-S(2,4)))
      S(1,4)=Y-S(1,2)
      S(2,1)=S(2,1)+S(1,4)**2
      IF(SENSE SWITCH3)45,6
   45 PRINT,X,Y,S(1,2),S(1,4)
      GO TO 6
   49 IF(M-N)52,53,52
```

```
   53 S(2,2)=SQR(S(2,1)/(S(1,3)-3.))
      GO TO 1045
   60 IF(SENSE SWITCH1)46,3
   52 PRINT,N,M,I
      PAUSE
   61 IF(SENSE SWITCH3)54,3
   46 L=4
      GO TO 6
C  HERE BEGINS PROGRAM FOR EXPONENTIAL (SEMI-LOG LINE) FIT, Y=A*(B**X)
 1001 DO1002K=1,2
      DO1002J=1,4
 1002 S(K,J)=0.
      PRINT,
      PRINT,2
      PRINT,
      MM=1
      IF(SENSE SWITCH2)1029,1006
 1006 L=5
      GO TO 6
 1007 IF(I)1018,1008,1008
 1008 S(1,3)=S(1,3)+1.
 9999 Y=LOG(Y)
      S(1,2)=S(1,2)+X
      S(2,2)=S(2,2)+X**2
      S(1,4)=S(1,4)+Y
      S(2,4)=S(2,4)+X*Y
      GO TO 6
 1018 N=S(1,3)
 S(2,1)=(S(1,2)*S(1,4)-S(1,3)*S(2,4))/(S(1,2)**2-S(1,3)*S(2,2))
      S(1,1)=(S(1,4)-S(2,1)*S(1,2))/S(1,3)
      S(2,4)=EXP(S(1,1))
      I=-1
      GO TO (1019,2019),MM
 1019 S(2,3)=EXP(S(2,1))
      PRINT,I,S(2,4),S(2,3)
      PRINT,S(2,4),S(2,1)
      PRINT,
      IF(SENSE SWITCH4)1001,1130
 1130 L=6
 1131 S(2,1)=0.
      M=0
      GO TO 6
 1032 IF(I)1843,1036,1036
 1036 M=M+1
      S(1,2)=S(2,4)*(S(2,3)**X)
      S(1,4)=Y-S(1,2)
      S(2,1)=S(2,1)+S(1,4)**2
      IF(SENSE SWITCH3)1039,6
 1039 PRINT,X,Y,S(1,2),S(1,4)
      GO TO 6
 1843 IF(M-N)1047,1043,1047
 1043 S(2,2)=SQR(S(2,1)/(S(1,3)-2.))
 1045 I=-1
      PRINT,N,S(2,2),I,S(2,1)
      PRINT,
      GO TO (1044,2044,60),MM
 1044 IF(SENSE SWITCH1)1052,1001
 1047 PRINT,N,M,I
      PAUSE
      GO TO (1048,2048,61),MM
 1048 IF(SENSE SWITCH3)1130,1001
 1029 L=7
 1030 ACCEPT,S(2,4),S(2,3)
      GO TO 6
 1129 IF(I)1001,1329,1329
 1329 S(1.2)=S(2.4)*(S(2.3)**X)
```



```
 1429 PRINT,X,S(I,2)
      GO TO 6
 1052 L=8
      GO TO 6
C  HERE BEGINS PROGRAM TO FIT LINEAR LOG-LOG PLOT FOR CURVE Y=A*(X**B)
 2001 DO2002K=1,2
      DO2002J=1,4
 2002 S(K,J)=0.
      PRINT,
      PRINT,3
      PRINT,
      MM=2
      IF(SENSE SWITCH2)2029,2006
 2006 L=9
      GO TO 6
 2007 IF(I)1018,2008,2008
 2008 S(1,3)=S(1,3)+1.
      X=LOG(X)
      GO TO 9999
 2019 S(2,3)=S(2,1)
      PRINT,I,S(2,4),S(2,1)
      PRINT,
      IF(SENSE SWITCH4)2001,2130
 2130 L=10
      GO TO 1131
 2032 IF(I)1843,2035,2035
 2035 M=M+1
      S(1,2)=S(2,4)*(X**S(2,3))
      S(1,4)=Y-S(1,2)
      S(2,1)=S(2,1)+S(1,4)**2
      IF(SENSE SWITCH3)1039,6
 2044 IF(SENSE SWITCH1)2052,2001
 2048 IF(SENSE SWITCH3)1131,2001
 2029 L=11
      GO TO 1030
 2129 IF(I)2001,2329,2329
 2329 S(1,2)=S(2,4)*(X**S(2,3))
      GO TO 1429
 2052 L=12
      GO TO 6
      END

END OF COMPILATION
LOAD SUBROUTINE DECK
THEN PUSH START
```

SYMBOL TABLE PRIOR TO SPREAD (INCOMPLETE)

This listing was obtained by using the 1-per-card program in the following manner: The last card in the punch stacker at time subroutines called for was (see below)

```
36200000010048RS‡
3818860
```

```
000000000000000000000000000012T35320232900000001TT
358802052T32160203500000000010T312802130T29560200800
00000009T286802006T344402029T262002002T259602001000
0000008T24720T329T23840T030000000007T342402048T234
00T048T25640T052T340402044T21800T0445120000000T1932
0T043T22000T047T17960T039T16080T036T18640T843000000
0000000000006T14720T130T302402019T13320T019T070809
999T06720T008T09080T0180000000005T05840T006T23600T0
29T03360T002T03120T0010000000004T0260000061T02800004
6T0180000060T19920T0455130000000T0112000053T020000052
0997600004554000000000977600050T004400049T14960T1310
96880000545500000000000000000600000000300000000500
00000004427273000042727100000000000000142727200000000
00000005110000000081960000808576000010T347602129T316
002032T290002007T24160T129T15520T032T06160T00709720
000510814000002680000000T25200T42900000000002670000
00000789600000449000000000784000007079880000600000000
0025300000000430000000042000000000796400001207772000
01000000000354540000000000000000T0000000000510000000
0520000000007524000005075000000300002T98096200000000
41635546004163550000435662460043566200006249554600 6
249550000625859460062585900005356474600535647000045
675746004567570000
```

noted. Compilation was then completed. One-per-card program was then loaded as far as card noted above. Appropriate place for type-out was determined, and this listing was produced. Purpose was to determine beginning of exponential and power subprograms which are underlined.

MAINTENANCE

LIKELY CHANGES

Programmed CS control should give sufficient versatility to meet most needs for these curves, especially at installations where machine configuration is compatible with that for which this program was designed. Source program cards numbered serially in columns 78-80 are furnished. Most likely changes will be to make I/O compatible with user's needs.

However, should alteration after compilation prove desirable, the last page of PROGRAM LISTING is typeout of most of symbol table prior to spread. Details are contained on that page.

In this regard, compressed program (p. 26) is more easily altered, as an absolute address appears in columns 1-5 of each card.

APPENDIX

BIBLIOGRAPHY

Graves, W. R., Polynomial Curve Fitting, (1961), IBM 1620 Users Group Library, White Plains, New York.

Hoel, Paul G., Introduction to Mathematical Statistics, (1954), John Wiley and Sons, New York.

Moore, Jim, IBM 1620 FORTRAN Compressor and 75-Column Dump, (1962), IBM, Birmingham, Alabama, (unpublished).

Rider, P., An Introduction to Modern Statistical Methods, (1939), John Wiley and Sons, New York.

Scarborough, James B., Numerical Mathematical Analysis, (1955), The John Hopkins Press, Baltimore, Maryland.

NOTES

1. The symbol $\overline{ZZ}$ in the writeup is equivalent to ZZ in the FORTRAN listing and was so written to eliminate any confusion arising from ZZ incorrectly implying a product to some readers.

2. When parameters or variables, a, b, etc., appear in text, they are underlined for reading ease. When they appear in equations or parentheses, they are not.

3. RS means perform release and start operations at 1620 console.

4. AN INTERPRETATION OF THE STANDARD ERROR OTHER THAN TESTING THE SIGNIFICANCE OF AN ADDITIONAL PARAMETER: This discussion also pertains to polynomial curve fitting, where use of the standard error in testing additional parameters is common. See Program Number 7.0.002 (Graves).

   The given empirical data quite likely do not lie on any single plane curve, because they are not the function of two but several variables.

   However, if the data represent the function of one main independent variable with the variability in the dependent variable due to lesser important independent variables, the curve fitting may possibly be treated statistically.

   An extremely useful assumption (provided it is valid, of course) is that the given data represent a random sample of y's from a normal population Y which has constant sigma square about the computed value y_c given by $y_c = f(x)$ where f(x) is the fitted function. It is useful because of its ease of computation and ease of interpretation.

   The assumption of constant sigma square for Y irrespective of x is in particular the assumption that errors of measurement and the effects of other independent variables of considerably less importance than the main one yield an aggregate that is normally distributed with constant sigma square, and independent of x.

   Describing this latter variability then is the real issue.

   What we would like to say is that approximately 95% of the population of values which the true function of several variables will take on lie between y_c -2sigma and y_c + 2sigma for all x. That is instead of fitting only the single curve y_c, we would fit in reality the family of curves $y_c \pm$ k(sigma), which have statistical significance and for which the standard error is a valuable estimate of sigma.

   Our question then is, "Is the variability of the less important variables normally distributed with constant sigma square?" If so, we can make a reasonable assumption that it is true for y in Y. Although the validity of such an assumption is beyond the scope of this writeup, it should be based upon more than the desire of the user for "nice" results.

   When statistical reliability is not needed, the sum of the deviations squared can be studied directly. This value is also furnished.

5. When fitting a curve to points, order in which points are given to 1620 does not matter. Order on Pass 1 need not be same as for Pass 2. However, if standard error desired, Pass 2 must use all points.

6. Numerical analysts are in general agreement that when a curve is fitted to empirical data, its use for extrapolation is to be approached with great caution. Interpolation is much more reliable.

7. This program may be compressed to 203 cards by use of General Purpose Dump and FORTRAN Compressor written by Jim Moore, IBM, Birmingham. Use of the Compressor (Appendix C) requires Indirect Addressing.

8. If manual branching from subprogram to subprogram occurs only at points indicated in operating instructions, result is to "frame" any work done by given subprogram between typings of its subprogram code number.

9. With reference to console switch settings: The notation XXXX (X = 0, 1) refers to CS1, CS2, CS3, CS4 from high to low order digit, respectively; 0 means OFF, 1 ON.

10. Similar treatment could be given to the general logarithmic equation $(y = b(\ln x) + \ln a)$, several other forms of hyperbola, and several forms for the circle and ellipse. Another possibility (requiring solution of a 6x6 system) is to fit a curve of the form $ax^2 + bxy + cy^2 + mx + ny + k = 0$, the general second degree curve. A 40K program similar to this 20K program could include this one and contain several other such subprograms. Likely the general equation would require a separate program, but still other subprograms from the list above could be packed into a second 40K program. Each curve type in this 20K program and in the above list has the advantage of tending not to be too sophisticated for ordinary empirical data to 3 or 4 significant figures. Southern Services plans to develop programs for these other curves as the need arises.

11. This program leaves unutilized approximately 5200 of the basic 20,000 core locations. This should be more than sufficient to permit addition to this 20K program of one more three-parameter subprogram of the user's choice.

COMPRESSES FORTRAN DECKS TO LESS THAN HALF ORIGINAL SIZE. EVEN COMPRESSED SPS DECKS MAY BENEFIT. ALLOWS SNAPSHOTS OF MEMORY OVER ANY INTERVAL. ROUTINE AND DUMPED DATA ARE SELF-LOADING WITH AUTOMATIC BRANCH TO PROGRAM. TABLES AND PROGRAM OVER POSITION 100 ARE INTACT IMMEDIATELY AFTER DUMPING AND ALSO UPON RELOADING. REQUIRES IDA.

INSURE COLUMNS 56-60 ON CARD 10 CONTAIN HIGHEST 1620 ADDRESS WITH FLAG. LOAD. INSERT DUMP STARTING ADDRESS WITH FLAG. R-S. INSERT LAST ADDRESS TO BE INCLUDED WITH FLAG. R-S. INSERT PROGRAM STARTING ADDRESS. R-S. INSERTS ARE OPTIONAL WITH 400, X9999, AND 7500 ASSUMED AS LIMITS. IN ANY EVENT POSITIONS 400 AND 401 ARE INCLUDED TO RETAIN RECORD MARKS, IF ANY. COLUMNS 1-5 ON DUMPED CARDS DEPICT LOCATION OF DATUM IN COLUMN 6. DUMPS WHETHER OR NOT TABLES ARE IN MEMORY.

FORTRAN USERS SHOULD DUMP TWICE IF THERE IS MUCH ROOM BETWEEN THE PROGRAM AND THE SYMBOL TABLES, NOTING THE BOUNDS OF EACH. THE TWO DECKS ARE THEN COMBINED REMOVING THE 4 REAR CARDS FROM THE FIRST DECK AND 5 FRONT CARDS FROM THE SECOND DECK. OTHERWISE, DUMP ALL OF MEMORY AND REMOVE CARDS WITH ZEROS FROM COLUMN 6 TO 80.

TO ELIMINATE THE HALT BETWEEN LOADING AND PROGRAM START CHANGE THE HALT COMMAND ON THE THIRD CARD FROM THE REAR TO NOP AND THE ADJACENT ALFA DOUBLE DIGIT REPRESENTATION FROM GO TO GONE WITH ALFA RECORD MARK.

NUMERIC BLANKS WILL BE DUMPED AS BLANKS. LOCATE SAME AND PUNCH WITH 4-8.

```
36000 0005003600300005003800300004003600300005003800300004003600320005003800322 0
04003600000005004900000000000000000000000000000000000000000000000000000000000000
36003000050036003220050036000800050049000920000000000000000000000000000000000000
01234567891234567890230000000000000000000000000000000000000000000000000000000000
23456789013456789012456789012356789012346789012345789012345689012345679012345678
36002200050038002200040036002200050038002200040036002200050049002200000000000000
00004900000036000000050049000000000000000000000000000000000000000000000000000000
36002200050016000350022525002240022511002240000111000350000143000240003441000000
36000000050036000800050036001600050049000000000000000000000000000000000000000000
36000790010036000550010012000590007426002240008314002243999946001520130025002250
14001100078000011100083000014300072000773800220004001600078002254900036026002240
00592600170000593600080005003600000005004900000000000000000000000000000000000000
05004900060000000000000000000000000000000000000000000000000000000000000000000000
25002250017011000060000111001700000143000000000538002200040049001880000000000000
36001400050038001400040036002200050038002200040036002200050038002200040036000000
00145021610050015102006021814200704112820080614223009081726300000000005060708090
00072360022000500360000000500490000000000000001020304000204060800030609021004080 2
01214161815181124272024282236352035304540363248445532494653604846546275445362718
00000000000000000000000000000000000000000000000000000036000800050036000000
05003600038001003800000004003600000005004900000000000000000000000000000000000000
340000000102390002700100484756 0‡000490750000000000000000000000000000000000000000
41000000000036000800050016000130004848000000000000000000000000000000000000000000
00000000000000000000000000000000010203040002040608000306090210040802161005001510 2
```

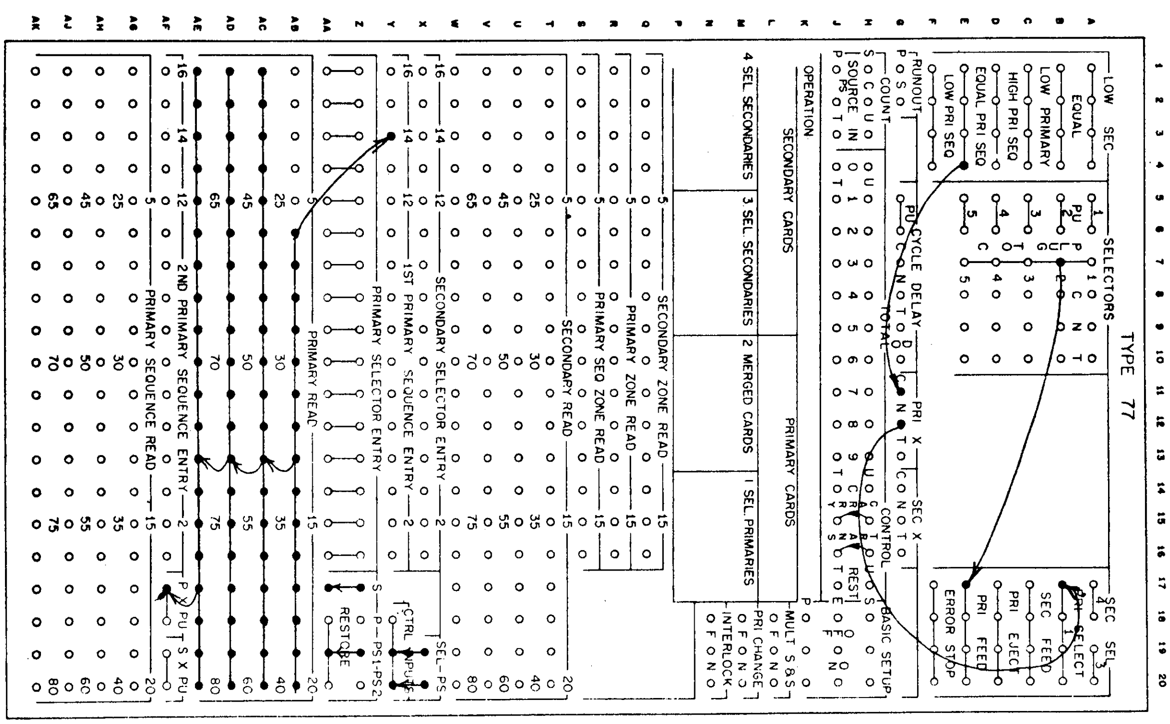
SEE PAGE 2 FOR 077 COLLATOR BOARD THAT WILL PULL ZERO CARDS.

PAGE 1 of 2

IBM 1620 FORTRAN COMPRESSOR AND 75 COLUMN DUMP
JIM MOORE, IBM, BIRMINGHAM

27

IBM
INTERNATIONAL BUSINESS MACHINES CORPORATION
COLLATOR, TYPE 77 CONTROL PANEL



28

NAME 1620 Dump DEPT

USE Remove All Cards with zero, Col 6 thru 80 NO

Page 2 of 2

COMPUTER
TECHNOLOGY