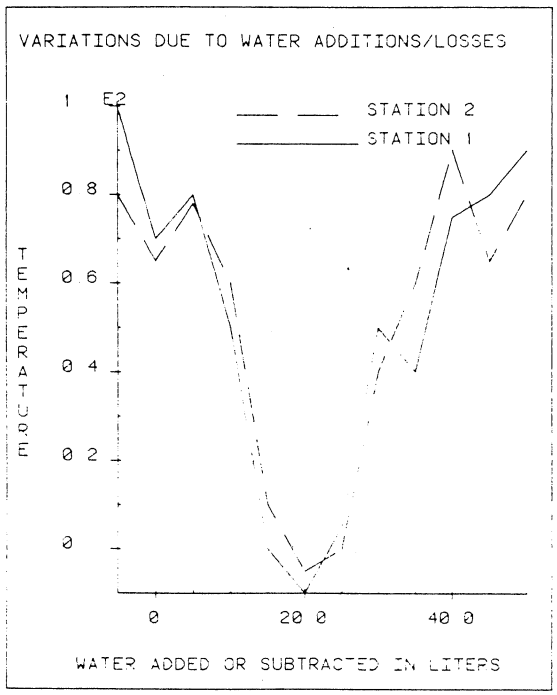
Tekniques

a Tektronix applications newsletter published monthly Vol. 1 No. 6



Producing this graph required many calculations. The adjacent article has a method for storing the graph data for quick redraw.

Solve Nonlinear Systems Of Equations In Seconds

By Gary P. Laroff

(This is the third in a series of articles on applications of the new Tektronix software package 4050A10 Statistics Volume 4.)

Equations that we are used to solving with pencil and paper are most often "linear" equations. Unfortunately many of the equations encountered in day to day experiences describe "nonlinear" phenomena, and are not so easily handled. The roots of a nonlinear equation f(x) = 0 cannot in general be expressed in closed form. In order to solve nonlinear equations, we are thus obliged to use approximate methods. These methods, normally presented for a single nonlinear equation, can be generalized to systems of nonlinear equations.

Consider the text book problem[1]:

...The system of equations

$$x = 1 + h^2(e^{x \cdot x} + 3x^2)$$

$$y = 0.5 + h^2 \tan(e^x + y^2)$$

(Continued on page 6.)

Storing A Graph On Tape From The 4051

By Dan Taylor

Some 4051 applications require a large number of calculations to produce a graph (for example, a 3-D surface). It would be nice to be able to store the finished graph on tape and get it quickly redrawn (on the screen or plotter) at a later time. This can be done in several different ways with the 4051 and its internal tape drive.

The most straightforward method of storing a graph is to add new lines of code after each MOVE and DRAW to write (in binary) the X-Y coordinates on tape. But storing only the X-Y values is not sufficient because you wouldn't be able to tell the MOVEs from the DRAWs on the tape. A solution is to store a third number with each X-Y pair. The method I use is:

1,X,Y	for a MOVE to X,Y
2,X,Y	for a DRAW to X,Y

This technique can then be extended to the other graphic commands:

3,N,X1,....,XN,Y1,....,YN	Array DRAW of N points
4,X,Y	RMOVE
5,X,Y	RDRAW
6,N,X1,....,XN,Y1,....,YN	Array RDRAW of N points
7	AXIS - no arguments
8,A,B	AXIS - 2 arguments
9,A,B,C,D	AXIS - 4 arguments
10,N,A$	PRINT the string A$ of length N
11,A	PRINT the number A
12,A,N,A$	PRINT USING A$: A (length A$=N)
13,A	ROTATE A
14,A,B,C,D	WINDOW A,B,C,D
15,A,B,C,D	VIEWPORT A,B,C,D
16,A,B	SCALE A,B

A simple BASIC program with graphics is:

```
100 VIEWPORT 30,100,20,80
110 WINDOW -2,10,0,300
120 AXIS 2,50
130 MOVE 2,100
140 DRAW 6,200
```

(Continued on page 4.)

In This Issue

* Editor's Note

New 4051 Applications Library Catalog Included With This Issue

The 4051 Applications Library is now 8 months old. In that time we've collected 59 programs for business, engineering, scientific, educational and general utility applications. A number of these programs have come from members of our 4051 Applications Library members, and your support is greatly appreciated. The wide variety of 4051 applications demonstrated by these programs include making super 8 animated films, petroleum risk analysis, schematic drafting, economic analysis, educational demonstrations, as well as the complex mathematical analysis and graphing capability, which is the 4051's specialty.

With this issue of Tekniques is a 12 page catalog of the programs collected to date by the 4051 Applications Library. This catalog includes reprints of the program abstracts that have appeared in the past issues of Tekniques, an index of the programs, and ordering information. All programs are available in our software exchange program. The catalog will be updated periodically. Additional copies of the catalog, are available from your local Tektronix Sales Engineer, or by writing Tektronix, Inc. P.O. Box 500, Group 451, Beaverton, Oregon 97077 c/o Applications Library. (60-369)

Besides developing a library of software for the 4051, we also see the 4051 Applications Library as our most effective means of interfacing with you the 4051 user. Submitting a program to the 4051 Applications Library not only allows you to share your efforts and knowledge of the 4051 with other 4051 users, but it also gives us a better idea of how we can meet your needs in the future. Even if you don't have a program to submit, we are always interested in hearing how you are putting the 4051 to work for you.

* * *

Tekniques, the 4051 Applications Library Newsletter, is published monthly by the Information Display Group of Tektronix, Inc., P.O. Box 500, Group 451, Beaverton, Oregon 97077. It is distributed to TEKTRONIX 4051 users and members of the 4051 Application Library.

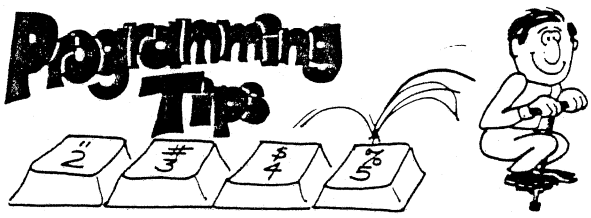
Publisher	**Ken Cramer**
Managing Editor	**Patricia Kelley**
Editor	**David Schwabe**

Volume 1 Number 6.

Do You Have A Friend Who Isn't A 4051 Applications Library Member Yet?

We're including an application card for membership in the 4501 Applications Library in this month's Tekniques. If you haven't sent us an application card yet, please do so, or give the card to a fellow 4051 user who would also like to receive Tekniques.

In addition to receiving Tekniques, the 4051 Applications Library newsletter, as a member of the 4051 Applications Library you are entitled to participate in the 4051 software exchange program. Submit one 4051 program to the 4051 Applications Library, and (if it is accepted) receive three programs of your choice from the library for free.



An Unusual Use of MOVE and DRAW

By Dan Taylor

In normal usage, the key word MOVE transforms coordinates from user units (window units) to graphic display units (GDUs). If the GDU location is off screen, nothing happens. If the GDU location is on screen, then a move (with the beam off) is done to the point, even if it is outside the viewport.

DRAW also transforms user units to GDUs. A vector is then drawn toward the point. If the point is outside the viewport, the vector is stopped (clipped) at the viewport boundary.

The simplest code for a draw and a move is equivalent to:

```
DRAW @32,20:
MOVE @32,21:
```

Where 32 is the address of the screen, 20 tells the screen to move with the beam on; and 21 tells the screen to move with the beam off.

The screen (or plotter) doesn't know what keyword (MOVE or DRAW) was called, only whether to move with the beam on or off. You can thus call a MOVE @32,20: to draw a vector that goes outside the viewport. Similarly, you can call DRAW @32,21: if you want a move (with the beam off) clipped at the viewport boundary.

GOTO And GOSUB Extend Branching Capability

By Ken Cramer

If you are an ex-FORTRAN programmer, some of the BASIC constructs in the 4051 may seem limited, particularly the branching associated with the true and false consequences of IF statements. The computed GOTO (GOTO n OF 500,600,700...), however, in 4051 BASIC allows extended branching capabilities similar to FORTRAN.

The FORTRAN IF syntax allows branching to three different statement numbers as a consequence of a minus, zero, or positive result. When using the 4051, the FORTRAN statement:

IF(A) 100,200,300

can be rewritten in 4051 BASIC as:

```
200 GO TO SGN(A)+2 OF 500,600,700
500 REM  BRANCH GOES TO HERE IF A IS NEGATIVE
510 REM  .
530 REM  CONTENTS OF ROUTINE
540 REM  .
590 GO TO 1000
600 REM  BRANCH GOES HERE IF A IS EXACTLY ZERO
610 REM  .
630 REM  CONTENTS OF ROUTINE
690 GO TO 1000
700 REM  BRANCH GOES HERE IF A IS POSITIVE
710 REM  .
730 REM  CONTENTS OF ROUTINE
740 REM  .
790 GO TO 1000
```

If A is an expression or A is a computed value, the chances of A being exactly zero are pretty small. To bring the FUZZ factor into the evaluation of A for a negative, zero, or positive result, use the following statement:

```
200 GO TO SGN(A)*(A=0)+2 OF 500,600,700
500 REM  BRANCH GOES HERE IF A IS NEGATIVE
510 REM  .
530 REM  CONTENTS OF ROUTINE
540 REM  .
590 GO TO 1000
600 REM  BRANCH GOES HERE IF A IS WITHIN FUZZ OF ZERO
610 REM  .
630 REM  CONTENTS OF ROUTINE
690 GO TO 1000
700 REM  BRANCH GOES HERE IF A IS POSITIVE
710 REM  .
730 REM  CONTENTS OF ROUTINE
740 REM  .
790 GO TO 1000
```

4051 BASIC expands the computed GOTO capability with the computed GOSUB statement. GOSUB is not available in FORTRAN. The FORTRAN statements:

IF (A=1) CALL SUB1
IF (A=2) CALL SUB2
IF (A=3) CALL SUB3

can be reduced in 4051 BASIC to one line of code:

```
200 GOSUB SGN(A)*(A=0)+2 OF 500,600,700
210 REM  EXECUTION OF ANY OF THE SUBROUTINES RETURNS HERE
220 REM  .
500 REM  BRANCH GOES TO HERE IF A IS NEGATIVE
510 REM  .
530 REM  CONTENTS OF ROUTINE
540 REM  .
590 RETURN
600 REM  BRANCH GOES HERE IF A IS WITHIN FUZZ OF ZERO
610 REM  .
630 REM  CONTENTS OF ROUTINE
690 RETURN
700 REM  BRANCH GOES HERE IF A IS POSITIVE
710 REM  .
730 REM  CONTENTS OF ROUTINE
740 REM  .
790 RETURN
```

When you want to execute a subroutine on a true or false condition, a dummy RETURN statement can be used to effect the same result. For example, the following FORTRAN operation:

IF (A < 0) CALL SUB 1

can be accomplished in 4051 BASIC as

```
200 GOSUB (A<0)+1 OF 500,600
210 REM  THE SUBROUTINE RETURNS HERE
220 REM  .
230 REM  .
500 REM  THE SUBROUTINE STARTS HERE
510 REM  .
520 REM  .
590 REM  THE FOLLOWING RETURN STATEMENT IS ALWAYS EXECUTED
595 REM      REGARDLESS OF A'S VALUE
600 RETURN
```

Easy Cyclical Counters

By LeRoy Nollette

It is often necessary to have a counter in your program that counts up to a certain number, then starts over. A day of the week counter, which counts up to 7 then starts over, is one example. The simplest way to write such a counter is:

```
100 W=0
110 W=W*(W<7)+1

.
.

160 GO TO 110
```

When W in line 110 is less than 7, it is increased by 1. When W is equal to 7 it becomes 1. This counter can be written to count to other numbers merely by changing the 7 to the desired upper limit.

GPIB Manual Now Available

The **GPIB Application Support** manual is now available through your Tektronix Sales Engineer. The part number to order is 070-2307-00 and the price is $20.00. This 250 page manual discusses the IEEE standard, the signals on the bus, data transmission and the GPIB handshake. Major sections of the manual are dedicated to program examples using Tektronix, Dana, Fluke, Wavetek and Hewlett-Packard equipment.

This manual will be useful to everyone interested in using the 4051 GPIB to interface to IEEE-488 equipment built by other companies.

Storing A Graph . . . . . from page 1

To store this graph on tape you would add the following lines to the program:

```
95 FIND J!
100 VIEWPORT 30,100,20,80
105 WRITE 15,30,100,20,80
110 WINDOW -2,10,0,300
115 WRITE 14,-2,10,0,300
120 AXIS 2,50
125 WRITE 8,20,50
130 MOVE 2,100
135 WRITE 1,2,100
140 DRAW 6,200
145 WRITE 2,6,100
147 CLOSE
```

Note that only the code numbers for the graphic commands and coordinates are stored on the tape. The calculations required to create these coordinates do not have to be performed again. The graph, or 3D surface can thus be redrawn very quickly from this information.

The following program can then be used to read such a tape file and reproduce the graph:

```
1000 PRINT "LRe-Draw a graph which is stored on tape."
1010 PRINT "JEnter the tape file # ";
1020 INPUT J
1030 IF J<>INT(J) OR J<1 OR J>255 THEN 1010
1040 PRINT "JEnter the device number for graphic output (screen=32) = ";
1050 INPUT D0
1060 IF D0<>INT(D0) OR D0<1 OR D0>33 THEN 1040
1070 FIND J
1080 ON EOF (0) THEN 4000
1090 PAGE
1100 REM  GET THE NEXT GRAPHIC ELEMENT
1110 READ @33:K
1120 GO TO K OF 2100,2200,2300,2400,2500,2600,2700,2800,2900,3000
1130 GO TO K-10 OF 3100,3200,3300,3400,3500,3600
1140 PRINT "LError:  This graph file is out of sequence."
1150 END
2090 REM  MOVE:
2100 READ @33:X0,Y0
2110 MOVE @D0:X0,Y0
2120 GO TO 1100
2190 REM  DRAW:
2200 READ @33:X0,Y0
2210 DRAW @D0:X0,Y0
2220 GO TO 1100
2290 REM  MATRIX DRAW:
2300 READ @33:N0
2310 DIM X1(N0),Y1(N0)
2320 READ @33:X1,Y1
2330 DRAW @D0:X1,Y1
2340 DELETE X1,Y1
2350 GO TO 1100
2390 REM  RMOVE:
2400 READ @33:X0,Y0
2410 RMOVE @D0:X0,Y0
2420 GO TO 1100
2490 REM  RDRAW:
2500 READ @33:X0,Y0
2510 RDRAW @D0:X0,Y0
2520 GO TO 1100
2590 REM  MATRIX RDRAW:
2600 READ @33:N0
2610 DIM X1(N0),Y1(N0)
2620 READ @33:X1,Y1
2630 RDRAW @D0:X1,Y1
2640 DELETE X1,Y1
2650 GO TO 1100
2690 REM  AXIS -- NO ARGUMENTS:
2700 AXIS @D0:
2710 GO TO 1100
2790 REM  AXIS -- 2 ARGUMENTS:
2800 READ @33:X0,Y0
2810 AXIS @D0:X0,Y0
2820 GO TO 1100
2890 REM  AXIS -- 4 ARGUMENTS:
2900 READ @33:X0,Y0,X2,Y2
2910 AXIS @D0:X0,Y0,X2,Y2
2920 GO TO 1100
2990 REM  PRINT A$; :
3000 READ @33:N0
3010 DIM A$(N0+1)
3020 READ @33:A$
3030 PRINT @D0:A$;
3040 DELETE A$
3050 GO TO 1100
3090 REM  PRINT A; :
3100 READ @33:X0
3110 PRINT @D0:X0;
3120 GO TO 1100
3190 REM  PRINT USING A$:A  :
```

The above technique is very general. There are a number of more restrictive methods of storing the graph on tape that can greatly increase the speed of the redrawing. One factor that slows redrawing is that the commands MOVE and DRAW X,Y must transform the X,Y values from user defined units into graphic display units (GDUs) and perform any clipping required at the VIEWPORT boundaries.

If your graph is produced with just the WINDOW, VIEWPORT, MOVE, DRAW and PRINT commands, **and no clipping** occurs, it is fairly simple to speed up the redrawing, by converting user coordinates to GDUs before they are stored on the tape.

If you use WINDOW W1,W2,W3,W4 and VIEWPORT P1,P2,P3,P4, then you can construct the following relationships to convert user defined coordinates to GDUs:

$$a = \frac{P2 - P1}{W2 - W1}$$

$$b = P1 - a*W1$$

$$c = \frac{P4 - P3}{W4 - W3}$$

$$d = P3 - c*W3$$

X and Y, in GDUs, is thus equal to:

$$X = a * x + b$$
$$Y = c * y + d$$

where x and y are in user defined units.

Note that MOVE x,y is now the same as PRINT @32,21: a * x + b, c * y + d, and DRAW x,y is the same as PRINT @32,20: a * x + b, c * y + d.

As with the first method, the graphic data can be written on tape in triplets, in the form:

20,X,Y	draw to X,Y	(X,Y are in GDU's, not user units)
21,X,Y	move to X,Y	
1,N,A$	print A$ of length N	
2,A	print the number A	
3,A,N,A$	print USING A$: A$ is of length N	

In writing a program to create this tape file, you use the relationships for the conversion factors a, b, c and d to convert coordinates from user defined units to GDUs. For example:

```
3200 READ @33:X0,N0
3210 DIM A$(N0+1)
3220 READ @33:A$
3230 PRINT @D0: USING A$:X0
3240 DELETE A$
3250 GO TO 1100
3290 REM  ROTATE:
3300 READ @33:X0
3310 ROTATE X0
3320 GO TO 1100
3390 REM  WINDOW:
3400 READ @33:X0,Y0,X2,Y2
3410 WINDOW X0,Y0,X2,Y2
3420 GO TO 1100
3490 REM  VIEWPORT:
3500 READ @33:X0,Y0,X2,Y2
3510 VIEWPORT X0,Y0,X2,Y2
3520 GO TO 1100
3590 REM  SCALE:
3600 READ @33:X0,Y0
3610 SCALE X0,Y0
3620 GO TO 1100
3990 REM  PICTURE FINISHED -- MOVE CURSOR TO UPPER RIGHT:
4000 INPUT @D0:X0,Y0
4010 PRINT @D0;21:X0,Y0
4020 HOME
4030 END
```

The program to read the resulting file from tape does all

the graphics with PRINT commands, using the parameters 20 and 21 to determine if the print is a move or a draw (see the 4051 Graphic System Reference Manual, pages 9-27 and 9-41, for the use of secondary addresses 20 and 21).

```
1100 READ @33:S
1110 GO TO S OF 3000,3100,3200
1120 REM  MOVE OR DRAW (GDU LEVEL):
1130 READ @33:X0,Y0
1140 PRINT @D0,S:X0,Y0
1150 GO TO 1100
```

It is several times faster than the first method. If your file doesn't have any PRINT commands this simplifies to:

```
1100 READ @33:S,X0,Y0
1110 PRINT @D0,S:X0,Y0
1120 GO TO 1100
```

The last technique is **another** two to **ten** times faster than the second method. It makes use of the fact that a graph normally consists of a move followed by a number of draws. The structure on tape is:

K,Z (for K $\geq$ 2 and even). K is the length of the **linear** array Z. And Z consists of alternating X and Y values - move to the first X,Y pair in Z and then draw to the rest. (X,Y are in GDU's)

-1,N,A$	Print, as previously
-2,A	Print, as previously
-3,A,N,A$	Print, as previously

A program to read such a tape file and produce a graph is:

```
1100 READ @33:K
1110 GO TO -K OF 3000,3100,3200
1120 REM  MATRIX "CURVE" (IN GDU):
1130 DELETE Z
1140 DIM Z(K)
1150 READ @33:Z
1160 PRINT @D0,21:Z(1),Z(2)
1170 IF K=2 THEN 1100
1180 PRINT @D0,20:Z
1190 GO TO 1100
```

When there are an average of 10 draws for every move this method is 7 times faster than the second method. The main difficulty with this last method is that your program to **create** the appropriate tape file becomes more complicated. (Because you have to know how many draws will follow before any can be put on tape.)

---

[1]The size that the tape should be marked depends upon how complicated your graph is. Each number in a binary file requires 10 bytes. Each string requires 3+ the number of characters in the string. A normal graph might have 100 draws (3 x 100 x 10 = 3000 bytes), therefore, using 10,000 byte files would be adequate for most graphs. (Binary files are used because they are faster.)

[2]Except for clipping i.e. this is true as long as W1 $\leq$ x $\leq$ W2 and W3 $\leq$ y $\leq$ W4.

More Information on Computer Animation Using the 4051

By Will Gallant

Last month's article by Mr. Paul Doherty of Oakland University in Rochester, Michigan on computer animation with the 4051 (see Tekniques Vol. 1 No. 5) presented some useful techniques for making super-8 animated films with the 4051. Several of us have been experimenting with computer animation and our after hours recreation in cinema photography has produced some notes we'd like to share with you. Like Mr. Doherty we've discovered that with some method of automatically triggering the camera single frame and some thoughtful programming, you can make animated films quickly and efficiently with a minimum of operator interaction.

A camera we use is the Minolta XL400. It has a single frame electronic shutter and a good *f*1.2 lens.

We use a different method of triggering the electronic shutter than suggested in the last issue of Tekniques. Pin 11 of the hard copy jack (J41) on the 4051 provides a 1/25 of a second trigger pulse whenever a COPY command is executed. The single frame input on the Minolta XL400 triggers when its input drops below 300 ohms. Fig. 1 shows the interfacing required to trigger your camera from the hard copy jack.

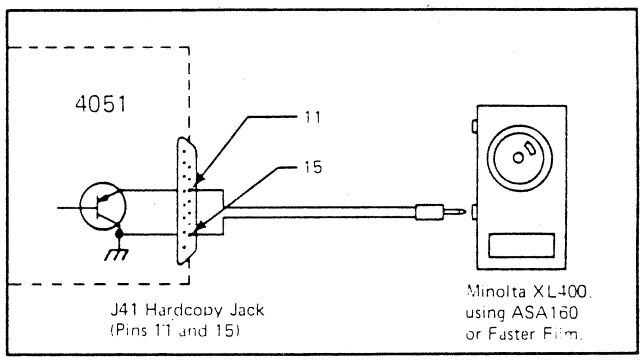


Fig. 1. Interfacing required to trigger camera's electronic single frame from hard copy jack.

Also as Mr. Doherty indicated, the resolution and brightness of the 4051 pushes low cost super 8 cameras and film to the limit. If you can tolerate less detail, movies made with Kodak ASA160 color film are adequate for projection in a very dark room. This film is readily available and adequate for most filming of graphic objects, however, text is very difficult to pick up clearly. GAF makes an ASA500 black and white film for super 8, but again you can expect some loss of detail in the bargain.

For maximum brightness of the movie, macro-focusing lenses are required, but they are hard to focus because of the small depth of field. One trick we learned is to hold a ruler perpendicular to the glass on the 4051 screen. The reflection of the ruler on the screen allows you to read the distance from the glass to the phosphor (Continued on page 8.)

Solve Nonlinear Equ. . . . . from page 1

can, for small values of h, be solved by iteration. Put $x_0 = 1$, $y_0 = -0.5$. These values are put into the right-hand side, where upon a new approximation (x_1, y_1) is obtained. The process is repeated until the changes in x and y are sufficiently small... solve the system for h = 0, 0.01, and 0.10.

Program 4 of Statistics Vol. 4, Nonlinear Systems Of Equations, can solve such problems, and is adaptable to the user's environment because it has been segmented into three parts:

1. A subroutine to solve the system of nonlinear equations,
2. a main program to use the subroutine, and
3. another subroutine, user written, that defines the system of equations. A sample subroutine is included in the initial program printout.

Method of Calculation

Powell's algorithm, used in Program 4, converges rapidly with nearly global convergence. The algorithm combines features of the Newton-Raphson, the Levenberg-Marquardt, and the steepest descent methods.

Nonlinear Systems of Equations (NLSE) requires N equations of the form $f_i(x_1, x_2 \ldots, x_N) = 0$. N represents the number of variables, and therefore the number of equations. j=1,2, ..., N; and $N \geq 1$. In vector form, $X=(x_1, \ldots, x_N)$, and $F(X) = (f_1(X), f_2(X), \ldots, f_N(X))$. If N=1, and x_k = value of x at kth iteration, Newton's method would be:

$$X_{k+1} = X_k - \frac{f(X_k)}{f'(X_k)} = X_k - (f'(X_k))^{-1} * f(X_k).$$

If $N \geq 2$, $X_{k+1} = X_k - J^{-1}(X_k) * F(X_k)$.

$J(X_k)$ is the Jacobian matrix of F, evaluated to X_k.

Variable DØ is used for approximating partial derivatives of the functions

$$\frac{\partial f_i}{\partial x_j} \approx (f_i(X+D\emptyset * e_j) - f_i(X))/D\emptyset$$

J_{ij} = partial of f_i with respect to x_j, evaluated at

$$X_k = \frac{\partial f_i}{\partial x_j}$$

Jij is approximated numerically by:

$$(f_i(X_k+D\emptyset * e_j) - f_i(X_k))/D\emptyset$$

where e_j is the normalized jth coordinate vector = a 0 vector except a 1 in the jth component.

Variable E represents required accuracy, and the algorithm has found a solution, X, when:

$$\sum_{j=1}^{N} (f_i(X))^2 \leq E.$$

The algorithm takes the sum of the squares of the f(X) values. When this sum is less than a specified E value, the program stops.

Using The Program

The user-written subroutine, starting at line number 1100, must define the system of equations and must end with a RETURN statement. Using the text book example above,

1. Rearrange equations to get all terms on the right-hand side:

$$0 = 1 + h^2(e^{xy} + 3x^2) - x$$

$$0 = 0.5 + h^2\tan(e^x + y^2) - y$$

2. Set the first equation equal to F(1), the second to F(2),....and the nth the F(n).
3. Replace the variables with x(i):

$$X(1) = x$$

$$X(2) = y$$

The equations now look like:

```
F(1)=1+H↑2*(EXP(X(2)*SQRX(1))+3*X(1)↑2)-X(1)

F(2)=0.5+H↑2*TAN(EXP(X(1))+X(2)↑2)-X(2)
```

Code these equations into the user-written subroutine starting at line 1100, as requested by the main program. Replace H in these equations with 0, 0.1, or 0.01 as you key in the equations.

For H = 0, the user-written subroutine looks like this:

```
1100 REM
1110 REM *-------------------------------------------------------------*
1120 REM *-------*SAMPLE USER WRITTEN SUBR. TO EVALUATE F AT X*-------*
1140 F(1)=1+0↑2*(EXP(X(2)*SQR(X(1)))+3*X(1)↑2)-X(1)
1150 F(2)=0.5+0↑2*TAN(EXP(X(1))+X(2)↑2)-X(2)
1160 RETURN
1170 REM *---------*END OF SAMPLE USER WRITTEN SUBROUTINE*-----------*
1180 REM *-------------------------------------------------------------*
1190 REM
```

For H = 0.1:

```
1100 REM
1110 REM *-------------------------------------------------------------*
1120 REM *-------*SAMPLE USER WRITTEN SUBR. TO EVALUATE F AT X*-------*
1140 F(1)=1+0.1↑2*(EXP(X(2)*SQR(X(1)))+3*X(1)↑2)-X(1)
1150 F(2)=0.5+0.1↑2*TAN(EXP(X(1))+X(2)↑2)-X(2)
1160 RETURN
1170 REM *---------*END OF SAMPLE USER WRITTEN SUBROUTINE*-----------*
1180 REM *-------------------------------------------------------------*
1190 REM
```

And for H = 0.01:

```
1100 REM
1110 REM *-------------------------------------------------------------*
1120 REM *-------*SAMPLE USER WRITTEN SUBR. TO EVALUATE F AT X*-------*
1140 F(1)=1+0.01↑2*(EXP(X(2)*SQR(X(1)))+3*X(1)↑2)-X(1)
1150 F(2)=0.5+0.01↑2*TAN(EXP(X(1))+X(2)↑2)-X(2)
1160 RETURN
1170 REM *---------*END OF SAMPLE USER WRITTEN SUBROUTINE*-----------*
1180 REM *-------------------------------------------------------------*
1190 REM
```

To solve for H = 0.01, key in RUN after the user-written subroutine, and enter the initial parameters for X(1) and X(2), 1 and 0.5 respectively. Then enter the desired accuracy. In this case the iterations are to be broken off when the sum of the squares of the function values are less than 10^{-8}.

```
                NONLINEAR SYSTEMS OF EQUATIONS

ENTER N, THE NUMBER OF EQUATIONS = 2
ENTER THE VECTOR X, THE INITIAL VALUE FOR ITERATIONS:
          X1 = 1
          X2 = .5
ENTER E, THE ACCURACY REQUIRED IN THE SOLUTION (EG, 1E-10) = 1E-8

ENTER D0, THE "DELTA" FOR THE DERIVATIVES (EG, 1E-3) = 1E-3
ENTER THE MAX. # OF ITERATIONS TO BE ALLOWED = 100
ENTER D1, AN UPPER BOUND ON THE DISTANCE TO A SOLUTION (EG, 100) = 100

DO YOU WANT X & F PRINTED AT EACH ITERATION STEP: YES
```

The time to solve the equations is 5 seconds, including the time to print out the intermediate results:

```
AFTER 1 EVALUTIONS OF F:
          J         X(J)                F(J)
          1         1                   4.6487212711E-4
          2         0.5                 -1.750671549E-5

AFTER 2 EVALUTIONS OF F:
          J         X(J)                F(J)
          1         1.001               -5.3448636E-4
          2         0.5                 -1.722654876E-5

AFTER 3 EVALUTIONS OF F:
          J         X(J)                F(J)
          1         1                   4.6503708171E-4
          2         0.501               -0.00101748356562

FINAL SOLUTION, AFTER 4 EVALUTIONS OF F IS:
          J         X(J)                F(J)
          1         1.00046516767       -7.289457926E-11
          2         0.499982621816      3.694822226E-13
```

Even with a choice of initial values of X(1) = 2 and X(2) = 1, the total time of the calculation is only 5 seconds. For H = 0, the computation time is well under 1 second.

In the next few months, look for part 4 on applications of Fourier Regressions to nonlinear function analysis.

---

[1]**Numerical Methods** by Germund Dahlquist and Ake Bjorck. Prentice-Hall, Englewood Cliffs, 1974, page 253.

4051 Applications Library Program Abstracts

Documentation and program listings of these programs may be ordered for $15.00 each. Programs will be put on tape for an additional $2.00 handling charge per program and a $26.00 charge for the tape cartridge. (The program material contained herein is supplied without warranty or representation of any kind. Tektronix, Inc. assumes no responsiblity and shall have no liability, consequential or otherwise, of any kind arising from the use of this program material or any part thereof.)

Please use the Applications Library Order Form. Order forms are included in the Membership Packet and are available from your local Tektronix Sales Engineer.

• SCIENTIFIC

ABSTRACT NUMBER: 51/00-5401/0

Title: **Graphics Demonstrations For Astronomy and Phsyics**
Author: Dr. R.J. Reimann, Physics Department, Boise State University
Memory Requirement: 8K
Peripherals: None

This is a program series, under the control of a directory, for the presentation of introductory astronomy and physics concepts to students with no computer experience. The series includes:

Sidereal Time—an aid to understanding.
Solar Time—a view of circumpolar constellations at any time.
Planetary motion—Circular orbits for the inner planets are plotted using Kepler's third law.
Binary Stars—circular orbits for binary systems are plotted using Kepler's third law.
Satellite Orbit—earth satellite orbits are plotted using a 2-step iterative approximation of Newton's law of universal gravitation.
Stellar Magnitudes—relative apparent magnitudes of stars are drawn assuming that brightness is proportional to area.
Phaser—a demonstration of the reference circle as related to simple harmonic motion.
Vector Combinations—addition of any number of vectors by the head-to-tail technique.
Trajectory With Drag—a quick 2-step iteration assuming drag is proportional to v^2.
Linear Least Squares Fit With Plot—best fit to a straight line is determined.

ABSTRACT NUMBER: 51/00-5701/0

Title: **Kaplan-Meier Survival Table Compu-Plotter**
Author: Paul W. Baim, Division of Biostatistics, University of Miami School of Medicine
Memory Requirement: 24K (Option 21)
Peripherals: 4631 Hard Copy Unit (Optional)

The survival table is a means of measuring the response or nonresponse of subjects over time. This response may mean improvement of patients receiving a new drug, or failure of light bulbs using a new type of filament; both lend themselves equally well to this type of study.

The program accepts raw data in the form of dates and subject status. The dates correspond to the beginning and end of the interval during which a particular subject is under study. The subject status indicates whether the subject is still responding at the end of the interval, or has stopped responding; for instance, whether a battery is still producing current after six days time, or has gone dead. The program then converts the raw data into a table of three columns: the interval, the cumulative survival, and the standard error. The interval is the same as that mentioned earlier, the cumulative survival represents the percentage of subjects surviving (i.e. responding) for this particular interval or longer, and the standard error establishes the reliability of the cumulative survival figure. Next, the program will, on demand, graph the calculated values, with the time intervals on the X axis and the cumulative survival percentages on the Y axis. The standard errors are shown by vertical lines. The purpose of the graph is to show clearly, how subjects are responding. Finally, the program will, on demand, list the data as it was entered.

• MISCELLANEOUS

ABSTRACT NUMBER: 51/00-6002/0

Title: **Print Mail Addresses and Form Letters**
Author: Nick Fkiaras
Memory Requirements: 24K (but 32K is recommended)
Peripherals: 4641

This package actually contains two programs.

Program One

This program allows the user to enter the text of a form letter and save it on file 3 of the same tape. The letter can have several pages, which are numbered automatically. The text may be edited by adding, inserting, deleting and replacing lines. Once the letter has been composed and

edited, it can be printed using the second program in this package.

The user may have the same salutation printed automatically for every addressee (entered and saved with the text of the letter) or he may enter it himself for each addressee as the letters are being printed.

Program Two

This program allows the user to enter any number of mail addresses, save them on tape automatically, then recall them later and have them printed on "Pin Feed Labels" with a Tektronix 4641 Printer. Each mail address can have up to five lines. Each line can be up to 32 characters long.

The user can edit his "mail address" files by deleting any number of mail addresses from them and by changing, replacing, inserting or deleting lines from each mail address. A permanent record of all files may be made by printing them on the 4641 printer using regular printer paper (132 columns).

Finally, the program will print form letters using the mail addresses and the text entered and saved by program one. More than one "mail address" file may be used.

• GRAPHICS

ABSTRACT NUMBER: 51/00-9501/0

Title: **Dash, Dot, Dash-Dot Routine**
Author: Nick Fkiaras
Memory Requirement: 8K
Peripherals: 4662 (Optional)

The program draws a dotted, dashed, or dot-dashed line between any two points X1,Y1 and X2,Y2 regardless of the window and viewport used. An example illustrating how to use the routine is included.

ABSTRACT NUMBER: 51/00-9513/0

Title: **Presentation Aids, Release 2**
Author: Will Gallant
Memory Requirement: 8K
Peripherals: 4662 Interactive Digital Plotter

This program is designed to aid in making 8 1/2 x 11 inch overhead transparencies. Text may be input to the 4051 and drawn on the 4662 Interactive Digital Plotter with full control over position and letter size. Special functions such as left and right justification, lines, boxes, and diamonds are also included.

A pie chart routine is also included in this program. Any number of segments may be assigned text and a value. The annotation is printed radially on the center line of its respective sector of the pie. The program will compute and annotate each segment in percentage form if desired. Further, any one segment of the chart can be exploded a specified distance from the pie for graphic emphasis.

ABSTRACT NUMBER: 51/00-9514/00

Title: **Pie Chart**
Author: P.C. Holman, University of Wisconsin—Stevens Point
Memory Requirement: 16K
Peripherals: 4631

Data to be presented as a pie chart is entered on the 4051 keyboard. The data is then presented as a table of original data, percent conversions, and degrees of a circle conversions. Data is automatically presented as a pie chart in one of two sizes, in one of seven screen locations. The program is tutorial from tape (ASCII).

Computer Animation . . . . . from page 5

(typically 3/4 inch). This can then be added to the distance from the glass to the lens.

If you can tolerate dimmer movies, it is much easier to place the camera back about 5 feet from the screen and zoom in. Your depth of field in this case increases to about 1-1/2 inches. For maximum brightness of the finished movie, the film should always be shot in a very dark room or construct a tube between the 4051 and the camera.

With the availability of the automatic shutter control, you should be able to write programs to create the graphic images to be filmed without the need to have an operator on hand. The following is a simple program that suggests the general approach. Note that after the COPY command in line 240, the screen is erased eight times. This erase loop is necessary because of the automatic origin shift in the 4051. It is recommended that this feature not be disabled. The total time to complete this erase loop is only 6 seconds, which doesn't add much to your per frame shooting time.

Here are a few methods of speeding up filming. When you make three dimensional transformations and rotations, the optional Matrix ROM pack may speed up calculation time as much as 3 to 10 times. You might also do your calculations before filming and store the graphics on tape. (See the article on page 1 of this issue of Tekniques titled "Storing Graphics From the 4051 On Tape".)

Another technique found to speed up filming is to take two frames of each graphic display, and then use larger increments of change between each transformation or rotation of the graphic image. You get the same rate of change of the image, but it takes half as long to film. Remember to wait a half second or so between COPY commands to allow your shutter to cycle. The following table gives you the filming and projection times for various intervals between frames.

Filming And Projection Times

Filming interval between successive frames in seconds.	Filming time required to produce 10 seconds projection time at 18 fps.	Time required to expose 50 feet of film.
.5	1.5 min.	
1	3 min.	1 hr.
2	6 min.	2 hr.
4	12 min.	4 hr.
8	24 min.	8 hr.
15	45 min.	15 hr.
30	1 hr. 30 min.	30 hr.
60	3 hr.	60 hr.