UNIVERSITY OF ILLINOIS

DIGITAL COMPUTER

ILLINOIS CODE F 1 - 114

TITLE	Solution of a System of Ordinary Differential Equations (Originally Code 27) (SADOI Only)
TYPE	Closed - with one program parameter

Location	Program	
p	00 mF	
	50 pF	The first word of
p+1	26 xF	Routine F 1 - 114 is at x.

NUMBER OF WORDS	41
TEMPORARY STORAGE	0, 1, 2, 3
PARAMETERS	The locations 3 to 7 must contain the following parameters before and during the input of this subroutine.

STORAGE LOCATIONS	CONTENTS	USE
3	00F 00 aF	N(a + i) are the variables y_i (i = 0, 1, ..., n - 1) Originally the initial values are placed here.
4	00F 00 bF	N(b + i) are the scaled derivatives, $2^m hy'_i$ $(=2^m hf_i)$, (i = 0, 1, 2, ..., n - 1), calculated by the auxiliary subroutine. $b > a + n-1$
5	00F 00 cF	Locations c + i, (i = 0, 1, ..., n-1) are used as temporary storage for this subroutine. These locations must be cleared to zero before this integration subroutine is entered for the first time. $c > b + n-1$.
6	00F 00 nF	n is the number of differential equations to be solved.
7	00F 00 dF	d is the location of the first word of the auxiliary subroutine.

DURATION

$T = 7 + n(15 + 0.1m) + 4t$ ms where

T = time in milliseconds to perform one step of integration.

t = time in milliseconds for the auxiliary subroutine.

DESCRIPTION

This subroutine will handle a set of n simultaneous first order ordinary differential equations, in which each derivative is expressed explicitly in terms of the variables

$$y_0' = f_0\ (y_0, y_1, \dots, y_{n-1})$$
$$y_1' = f_1\ (y_0, y_1, \dots, y_{n-1})$$
$$y_{n-1}' = f_{n-1}\ (y_0, y_1, \dots, y_{n-1})$$

Any differential equation or set of differential equations to be solved must first be expressed in the above form before this subroutine can be applied. For example, the second order differential equation

$$y'' = w^2 y$$

must be written as two first order differential equations

$$y_0' = wy_1 \qquad\qquad y_1' = wy_0$$

where $y_1 = y$ and $y_0 = y'/w$.

Each time this subroutine is called into use, it will carry out one integration step of length h. Each of the integrals y_i $(i = 0, 1, 2, \dots, n-1)$ is replaced by its value at the end of a step of length h. In doing so, this subroutine employs an auxiliary closed subroutine which evaluates the functions $f_0, f_1, f_2, \dots, f_{n-1}$ from the given values of y_i. The coder must write this auxiliary subroutine for his individual problem since it defines the equations being solved and, this depends entirely on his specific problem.

The purpose of the auxiliary subroutine is to calculate and store in locations $b + i$, $(i = 0, 1, 2, \dots, n-1)$, the quantities hf_i multiplied by a suitable scale factor 2^m. h is the increment of the independent variable and m is a positive integer to be chosen as large as possible without having any of the quantities $2^m hf_i$ exceed capacity anywhere throughout the range of integration. The factor 2^m is introduced to increase the accuracy of the integration subroutine. The variables, y_i, must all be scaled so that they are less than one throughout the range of integration before they are used in the auxiliary subroutine. Also for maximum accuracy, one should store $2^m h$ instead of just h. This auxiliary subroutine must be located in a sequence of locations beginning with location d, where d is defined by the parameter S7. In integrating over one step, the integration subroutine will call in the auxiliary subroutine four times.

This integration subroutine requires 3n arbitrary storage locations. The n consecutive locations a + i, (i = 0, 1, 2, ..., n-1; a arbitrary), are used to store the variables y_i. It is in these locations that the initial values are to be placed. It is also in these locations that the final results are found. The n consecutive locations b + i, (i = 0, 1, 2, ..., n-1; b > a + n-1), are used to store the scaled derivatives, $2^m h'_y$ ($= 2^m hf_i$), which are calculated by the auxiliary subroutine. The n consecutive locations c + i (i = 0, 1, 2, ..., n-1; c > b + n-1) are used for temporary storage by the integration subroutine. These locations will hold the quantities $2^m q_i$ (See page 7). The numbers left in these locations at the end of an integration step are $3{\cdot}2^m$ times the roundoff errors of the quantities y_i. These numbers are taken into account during the following step and serve to prevent the rapid accumulation of roundoff errors. As a result the effective numerical accuracy is m digits more than the capacity of the storage locations. Therefore, it is important that the locations c + i, (i = 0, 1, 2, ..., n-1) be cleared to zero before the integration subroutine is entered. Otherwise, this integration subroutine will add spurious corrections to the variables. Thus before the integration subroutine can be entered, the main routine must clear the temporary storage locations c + i to zero and set the initial values of the variables y_i in locations a + i.

SUMMARY

Supposing that, in the course of his routine, a coder has to solve a set of differential equations over a specified range given the initial value of the independent variable, a possible procedure would be the following:

(1) Reduce the given set of differential equations to a set of n first order differential equations.

(2) Calculate the initial values of the dependent variables, y_i.

(3) Scale all the functions so that all the values y_i are less than one throughout the range of integration.

(4) Choose a proper value of h (See note I).

(5) Choose m properly.

(6) Determine the parameters to be placed in S3 - S7, observing that $a < b < c$.

(7) Write an auxiliary subroutine which evaluates the functions $2^m hf_i$ and stores them in locations b + i.

(8) Make certain that the main routine sets the scaled initial values in locations a + i, and clears the temporary storage locations c + i to zero before the integration subroutine is entered.

With respect to the solution of a set of differential equations, a program can be broken up into four parts:

(1) Locations 3 through 7 which contain the parameters,

(2) The main routine,

(3) The integration subroutine (Code F 1 - 114)

(4) The auxiliary subroutine.

THE INDEPENDENT VARIABLE

If the independent variable x occurs in the functions f_i or if it is required during an integration as an index, then it must be obtained by integrating the equation $x' = 1$. The independent variable x is then treated as an additional dependent variable, for which the auxiliary subroutine has to provide the quantity $2^m hx' = 2^m h$. However, this latter quantity may be planted at the beginning of the integration in the appropriate location (e.g. in location b) and left there, so that the auxiliary subroutine is relieved of the task. If the independent variable does not appear in any of the f_i's but is merely wanted for indication purposes, it is quicker to use a simple counter in the main routine.

NOTES.

I) Accuracy: The truncation error in one step is of the order of h^5. Ordinarily, that is for a small set of well behaved equations, its magnitude is about $10^{-2}h^5$; for large sets or difficult equations it may be greater. Over the range of integration this error will amount to about $h^4/100$. Roundoff errors accumulate at a rate corresponding to the keeping of $(39 + m)$ binary digits. The choice of the length of the increment h is governed largely by the accuracy desired. An increase in the length of h will result in a decrease in accuracy and in operating time. Likewise, a decrease in the length of h will result in an increase in both accuracy and operating time. However, no further increase in accuracy can be gained by choosing $h < 2^{-8}$ because of the introduced truncation error. But, if the functions are very sensitive to variations in y_i, or if the number of equations is very large, smaller steps will probably be necessary with, of course, a corresponding increase in the time required. Now, the process used in the integration subroutine is a fourth order one. Thus, 1/15 of the following difference,

(the value of y_e calculated using an interval of length h)

-(the value of y_e calculated using an interval of length 2h)

is an approximation of the error.

II) Adjustment of the increment : There exist essentially two ways of adjusting the increment

a) One may double or halve the increment by varying the value of m in the main routine. This may be done over the complete range of integration or just over part of it. When only the parameter 00 mF in the link between the main routine and the auxiliary subroutine is changed to 00 (m+1)F and the auxiliary subroutine is unaltered, the length of the increment is halved. Likewise, when only the parameter 00 mF is changed to 00 (m-1)F the length of the increment is doubled. The auxiliary subroutine is not altered since $2^m h = 2^{m+1} h/2$. If one adjusts the increment over the complete range, adjusting only the value of m is sufficient. However, if one wishes to adjust the length of the increment within the range of integration, one must also adjust all the quantities in locations c + i. Otherwise, one will introduce roundoff errors in y_i of the magnitude,

$$2 \text{ (old value of h - new value of h)} \times 2^{-40}.$$

Now by also doubling the quantities in c + i when one halves the increment one will introduce no roundoff error. Similarly, by halving the quantities in c + i when one doubles the length of the increment, one will introduce no roundoff error. If one clears the locations c + i, one introduces roundoff errors of magnitude 2^{-40}.

b) One may alter the length of the increment in any ratio by adjusting the scaling factor $2^m h$ in the auxiliary subroutine. Here also one may adjust the length of the increment within the range of integration. Now it is not necessary to adjust the quantities in c + i. If, however, $2^m h$ becomes small, then roundoff errors are introduced by inaccuracies in the auxiliary subroutine. Thus one should not keep $2^m h$ small when integrating over large ranges unless the loss of accuracy and time does not matter.

III) Often it is desired to evaluate functions involving expressions like sin x or $J_m(x)$. These expressions can be evaluated by solving extra distinct differential equations along with the desired ones. For example,

$$d^2/dx^2 \quad (\sin x)/2 = -(\sin x)/2.$$

Thus we can evaluate (sin x)/2 by using the extra pair of equations

$$y'_{n+1} = 2^m h y_n \qquad\qquad y'_n = -2^m h y_{n+1}$$

and suitable initial conditions.

<u>METHOD USED FOR INTEGRATION IN THE ROUTINE</u>

Given a set of differential equations,

$$y'_i = f_i \ (y_0, y_1, y_2, \ldots, y_{n-1}), \quad (i = 0, 1, 2, \ldots, n-1)$$

The process used in the integration is defined by the following equations

$$k_{ij} = 2^m h f_i \ (y_{0j}, y_{1j}, \ldots, y_{n-1\ j})$$

$$r_{i,j+1} = (A_{j+1} + 1) \quad (k_{i,j} - B_j \ q_{i,j})$$

$$y_{i,j+1} = y_{i,j} + 2^{-m} \ r_{i,j+1}$$

$$q_{i,j+1} = q_{ij} + 3r_{i,j} + (C_j - 1) \ k_{i,j+1}$$

with the following table of values

j	A_{j+1}	B_j	C_j
0	-1/2	2	1/2
1	$-(1/2)^{1/2}$	1	$(1/2)^{1/2}$
2	$(1/2)^{1/2}$	1	$-(1/2)^{1/2}$
3	-5/6	2	1/2

Of the double subscripts used in the above equations, the first subscript, i, indicates which variable is being considered, and the second subscript, j, indicate which of the four parts of one step is being performed. The auxiliary subroutine evaluates the quantities $k_{i,j}$. In the above equations, only the quantities $q_{i,4}$ and $y_{i,4}$ are carried over from step to step. The quantities $r_{i,j}$ are calculated in the course of one step; they are not carried directly from step to step. When $j = 4$, we replace it by zero, increase i by 1, and terminate the step.

For one step, the sequence of operations is as follows:

$j = 0$	$i = 0, 1, 2, \ldots, n-1$
$j = 1$	$i = 0, 1, 2, \ldots, n-1$
$j = 2$	$i = 0, 1, 2, \ldots, n-1$
$j = 3$	$i = 0, 1, 2, \ldots, n-1$

REFERENCES

Gill, S., "A Process for the Step-by-Step Integration of Differential Equations in an Automatic Digital Computing Machine", Proceedings of the Cambridge Philosophical Society, vol. 47 (1951) pp. 96 - 108.

Wilkes, M. V., Wheeler, D.J., and Gill, S., The Preparations of Programs for an Electronic Digital Computer Addison-Wesley Press, Inc. Cambridge, Mass., (1951) pp. 32-33, 56-57, 86-87, 132-134.

DATE October 27, 1953 Rt. 3/19/59
CODED BY D.J. Wheeler
REVISED BY R. Polivka
APPROVED BY J.P. Nash

LOCATION	ORDER	NOTES	PAGE 1
	00 K(F1)		
	00 S3		
	00 S4		
	00 S5		
	00 S6		
	01 29K		
0	L5 5F		
	L4 6F		
1	00 20F		
	46 571F		
2	L5 4F		
	L0 3F		
3	42 573F		
	00 20F		
4	46 573F		
	L5 5F		
5	L0 4F		
	42 574F		
6	00 20F		
	46 574F		
7	26 93F		
	00 F		
8	64 F	c+n	
	00 33L		
9	80 S3	a	
	00 S3	a	
10	00 F	b-a	
	00.F	b-a	
11	00 F	c-b	
	00 F	c-b	
12	26 1469N		
	01 K		
0	S5 F	Set shift addresses = m	
	46 8L		
1	46 11L		
	L4 3L		

LOCATION	ORDER		NOTES
2	42 22L		Set link address
	22 21L		
3	L5 $(q_{i,j})$F	From 21	
	40 1F	By 20	
4	L5 $(k_{i,j})$F	By 18	
	40 F		
5	L0 (1)F	By 23	
	L0 1F		$(k_{i,j} - B_j q_{ij})$
6	40 3F		
	50 (A_j)F	By 24	
7	7J 3F		
	L4 3F		$(k_{ij} - B_j q_{ij}) (A_j + 1) 2^{-m}$
8	10 (m)F	By 0	
	40 3F		
9	L4 (y_{ij})F	By 17	Step y_i
	40 $(y_{i,j+1})$F	By 17	
10	L5 3F		
	50 2F		Form $r_{i,j+1}$
11	00 (m)F	By 1	
	40 3F		
12	50 (C_j)F	By 25	
	7J F		$C_j k_{ij} - k_{ij}$
13	L0 F		
	L4 1F		
14	L4 3F		
	L4 3F		$q_{ij} + 3 r_{i,j+1}$
15	L4 3F		
	40 $(q_{i,j+1})$F	By 19	
16	L5 9L		
	L4 13L		
17	42 9L		Increase all addresses depending
	46 9L		
18	L4 39L		
	46 4L		on i by 1

LOCATION	ORDER		NOTES PAGE 3 F1
19	L4 40L		
	42 15L		
20	46 3L		until i = n
	L0 37L		
21	36 3L		
	L5 (33)L	By 22,	
22	42 21L	From 2'	Increase j from 0 to 3 and then leave
	32 ( )F	By 2	by link
23	46 5L		
	10 10F		Adjust addresses which depend on j
24	L4 18L		
	42 6L		
25	46 12L		
	50 25L		Call in auxiliary subroutine
26	26 S7		
	41 2F		Clear 2F so that it can be used as zero.
27	L5 38L		
	26 17L		Start new i cycle.
28	40 F		
	00 F	1/2	C_0, C_3
29	NO F		
	00 F	-1/2	A_0
30	40 F		
	00 2071 0678 1186 J		$1/\sqrt{2}$ C_1, A_2
31	80 F		
	00 2928 9321 8814 J		$-1/\sqrt{2}$ A_1, C_2
32	80 F		
	00 1666 6666 6667 J		-5/6 A_3
33	LJ 1025F	-11, 1	
	06 1058L	25,34L	Expressed in units of 2^{-9}, 2^{-19}, 2^{-29}, 2^{-39}.
34	LJ 3074F	-9 2	Addresses used to set addresses to refer to
	06 3107L	27, 35L	the constants A_j, C_j and make the address in
35	LF 2F	-8, 2	5L, 1 or 2 according as B_j 2 or 1, and to
	06 2084L	26, 36L	stop the address in 21, dependent on j, and
36	LJ 1025F	-11, 1	to stop when positive.
	07 37L	28, 37L	
	0141K		