Fermilab Library
0 1160 0038515 7

MURA-201
Internal

MIDWESTERN UNIVERSITIES RESEARCH ASSOCIATION*

NUMERICAL TECHNIQUES

Harry Denman+

October 19, 1956

ABSTRACT: The most important MURA digital computer programs for studying particle orbits in the spiral sector FFAG accelerator are reviewed in some detail. Several suggestions are given for improving the speed and accuracy of these computations.

* Supported by Contract AEC #AT(11-1)-384

+ On leave from Wayne University - Summer Session 1956

Among the most important problems now being considered are solutions of the electromagnetic equations (Maxwell's equations) and the dynamical equations for the motions of charged particles in various proposed accelerators. The complexity of these equations and the lack of analytical solutions have resulted in the extensive use of numerical calculations to obtain approximate solutions for particular cases. The magnitude of these calculations has required the use of a large, high-speed, stored-program electronic computer. Up to the present, these calculations have been carried out almost exclusively on the Illiac computer. In the near future it is planned to use an IBM 704. In this report we shall discuss some of the programs already in use on the Illiac, certain numerical formulas and ideas which might be used, and changes and additions to these programs which might be feasible or preferable for the IBM 704.

At the present time the two most important programs used on the Illiac are:

1. Scapocyl, which produces (by the solution of Laplace's equation) the magnetostatic potential corresponding to a given pole and boundary conditions; the output of this program is a mesh of potential values corresponding to equally spaced points in the $\xi\eta$ plane.
2. Stormesh, which makes use of these stored potential values to obtain the field components by interpolation and differentiation; these field components are then used in the solution of the equations of motion for the orbit.

INTERPOLATION - DIFFERENTIATION FORMULAE

If the potential function $\Omega(\xi,\eta)$ or $\frac{\Omega}{\eta}$ is stored on a mesh ($\frac{\Omega}{\eta}$ is stored because it varies less rapidly than Ω itself) it must be differentiated in order to obtain the field components. These differentiations are handled differently for the ξ and η directions, if $\frac{\Omega}{\eta}$ is stored, since $\frac{\partial\Omega}{\partial\eta} = \eta\,\frac{\partial(\Omega/\eta)}{\partial\eta} + \frac{\Omega}{\eta}$.

Thus storage of $\frac{\Omega}{n}$ requires interpolation as well as differentiation-interpolation of the stored values to obtain the field components at points other than the mesh points.

A condition which has been imposed on the interpolation and interpolation-differentiation procedures is that they produce continuous values at the mesh points. The procedure which has been adopted to obtain the function and its derivatives inside a mesh interval is to apply a one-dimensional polynomial formula, first in one direction and then the other. It was thought that an interpolation formula involving the 4 points surrounding the interval in which the function was to be evaluated would give satisfactory results for the function.

To speed the computer program, it is desirable that the differentiation-interpolation formula be the derivative of the interpolation formula for the function itself. This is not necessarily the best formula for the derivative, but it is more convenient. Then a third degree interpolation polynomial for the function has the 4 conditions of continuity at the end points and continuity of its derivatives there also. This results in the formula, given in LJL 8, pg 7-8.

$$f(x_0+uh) = f_0 + \frac{u}{2}(f_1 - f_{-1}) + \frac{u^2}{2}(-f_2 + 4f_1 - 5f_0 + 2f_{-1}) + \frac{u^3}{2}(f_2 - 3f_1 + 3f_0 - f_{-1}) \quad \text{I}$$

which has as its derivative

$$hf'(x_0+uh) = \tfrac{1}{2}(f_1 - f_{-1}) + u(-f_2 + 4f_1 - 5f_0 + 2f_{-1}) + \tfrac{3}{2}u^2(f_2 - 3f_1 + 3f_0 - f_{-1}) \quad \text{II}$$

where $0 \leq u \leq 1$.

If one writes a general 2nd degree polynomial expression for $hf'(x_0+uh)$

$$hf'(x_0+uh) = a_0 + a_1 u + a_2 u^2$$

and requires continuity of this function at the mesh points, then $hf'(x_0) = a_0$ must not be a function of the mesh point value f_2. Then the best value for a_0 is $\frac{1}{2}(f_1 - f_{-1})$, but this has errors starting with the $\frac{d^3f}{dx^3}$ term, if one expresses f in terms of a power series about x_0. Thus the errors in the

formula II may be quite large, compared with other interpolation-differentiation formulas. A test of this formula for the function sin x has been made (this corresponds roughly to the variation of Ω with ξ), and is given in Table I. The sine curve here goes through one complete cycle in 50 mesh steps, and the interpolation and differentiation has been done at 9 points inside each of two mesh intervals, from 38° to 52°. The average absolute error is .000735 over this range, using formula II.

If one simply asks for an interpolation formula for $f'(x)$ of a certain order, involving certain mesh points, and also requires continuity of this formula, but does not require that this be the derivative of any particular interpolation formula for $f(x)$, then one may obtain other formulae for $f'(x)$ which are in some sense better than II. One such formula has been worked out by H. Meier and is

$$hf'(x_0+uh) = \frac{f_1-f_{-1}}{2} + \frac{u}{3}(-2f_2+9f_1-12f_0+5f_{-1}) + \frac{u^2}{6}(7f_2-21f_1+21f_0-7f_{-1}) \qquad \text{III}$$

where $0 \leq u \leq 1$

This formula also has been used to calculate $\frac{d\sin x}{dx}$ over the range $38\text{-}52^\circ$, giving the results labelled III in Table I. The average absolute error here is .000650, as compared with .000735 for formula II. The only disadvantage here is that III is not the derivative of a useful formula for interpolation of $f(x)$ itself, and therefore, if both interpolation and interpolation-differentiation must be done, formulae I and II used together are more efficient.

The term $\frac{f_1-f_{-1}}{2}$ in both formulae II and III is required for continuity in the interval with the mesh points used, but it contributes in large measure to the errors obtained in using these formulae. If one uses an interval which begins and ends at points midway between the mesh points, continuity requirements still lead to use of the adjacent mesh values for the first term in f', but now these points are much closer together. For a four-point formula, if we want continuity at 1/2 points and symmetry, we are led to the formula (covering 2 mesh intervals)

$$hf' = (f_0-f_{-1}) + \frac{u'}{12}(-7f_2+33f_1-45f_0+19f_{-1}) + \frac{u'^2}{24}(13f_2-39f_1+39f_0-13f_{-1}) \qquad \text{IV}$$

Here $0 \leq u' \leq 2$ corresponds to $-\frac{1}{2} \leq u \leq \frac{3}{2}$ in formulae II and III, and continuity is preserved at $u = -\frac{1}{2}$ and $\frac{3}{2}$. This formula, also derived by H. Meier, has the additional advantage of covering two mesh intervals, and therefore one might be able to do more than one Runge-Kutta step in the integration of the equations of motion within the same larger mesh region which this formula covers. Again, however, it is not the derivative of any very precise or continuous interpolation formula for f(x).

Discontinuous Derivative Formulae

If the requirement of continuity of the derivative is removed, one can obtain more accurate formulae. It turns out that one of those for f'(x) derivable as best from the Taylor series expansion is just the derivative of the Bessel interpolation formula

$$f = f_0 + \frac{u}{6}(-f_2 + 6f_1 - 3f_0 - 2f_{-1}) + \frac{u^2}{2}(f_1 - 2f_0 + f_{-1}) + \frac{u^3}{6}(f_2 - 3f_1 + 3f_0 - f_{-1}) \quad \text{V}$$

where $0 \leq u \leq 1$.

The derivative formula is then

$$hf'(x_0 + uh) = \frac{1}{6}(-f_2 + 6f_1 - 3f_0 - 2f_{-1}) + u(f_1 - 2f_0 + f_{-1}) + \frac{u^2}{2}(f_2 - 3f_1 + 3f_0 - f_{-1}) \quad \text{VI}$$

Applying this formula to f(x) = sin x, the average absolute error is reduced to .000062 (see Table I).

The requirement of continuity of the derivative on the mesh boundaries was considered necessary because if a group of particles occupying a certain continuous region of phase space were considered to pass this boundary, the discontinuity in the derivative might tend to shred the phase region, or cause the motion to appear unstable. However, when considering a finite group of particles, occupying a region of measure 0 in phase space, it is not clear that a discontinuity must be non-Liouvillian. However, it clearly might have some effect on the equations of motion, as this discontinuity behaves like a small bump, and the effects of bumps on the motion can be appreciable. The discontinuities, however, are quite small when the Bessel formula is used, and therefore, it was decided to try changing the Stormesh program to use formulas V and VI on a trial basis, and to

observe the effects on the particle motion. The Illiac program (overwrite) which accomplishes this change in the Stormesh program is given in Appendix I.

The modified Stormesh program was run on several sample potentials, and the results, for relatively short runs, were found to agree closely with those obtained previously, both with the Stormesh program and the Ridge-Runner. However, the old Stormesh results were obtained when 3 mesh values were stored in one Illiac register; this packing did produce some errors. Therefore, it has been decided to run further checks on this change to see if any significantly different results are obtained. Longer runs will also be attempted, with checks against the other Stormesh program as well as the Ridge-Runner program (for motion in the median plane). If z motion is included, the program cannot be checked against the Ridge-Runner. The Feckless Five program can handle z motion, but its accuracy is unknown. One can only guess whether an improvement has been made. If a previously stable motion case now exhibits instability, or vice versa, then a serious investigation into the relative correctness of these results should be made. Apparently, from the first results, however, neither the discontinuity nor the increased accuracy have made much difference in a short run.

The seriousness of the errors introduced by requiring continuity in the derivative of the interpolation formula involving 4 points raised the question of the suitability of a 4 point formula under such circumstances. A 3 point continuous derivative formula was found:

$$hf'(x_0+uh) = \frac{f_1 - f_{-1}}{2} + (f_1 - 2f_0 + f_{-1})u, \quad -\tfrac{1}{2} \le u \le \tfrac{1}{2} \qquad \text{VII}$$

which is continuous at the $\frac{1}{2}$ points. This formula was also tried on $f(x) = \sin x$, and the results given in column VII, Table I. The average absolute error is .001227, which is not much greater than for the continuous 4 point formulas.

If one goes to a formula involving 5 points, and thus of 4th degree,

one obtains the continuous interpolation formula

$$f(x_0+uh)=f_0+\frac{u}{12}(8f_1-8f_{-1}+f_{-2}-f_2)+\frac{u^2}{48}[68(f_1+f_{-1})-11(f_2+f_{-2})-114f_0]$$
$$+\frac{u^3}{12}(f_2-2f_1-f_{-2}+2f_{-1})+\frac{5u^4}{12}[6f_0-4(f_1+f_{-1})+f_2-f_{-2}], \text{ where } -\tfrac{1}{2}\le u\le\tfrac{1}{2}, \qquad \text{VIII}$$

which has the continuous derivative

$$hf'(x_0+uh)=\frac{1}{12}(8f_1-8f_{-1}+f_{-2}-f_2)+\frac{u}{24}[68(f_1+f_{-1})-11(f_2+f_{-2})-114f_0]$$
$$+\frac{u^2}{4}(f_2-2f_1-f_{-2}+2f_{-1})+\frac{5}{3}u^3[6f_0-4(f_1+f_{-1})+f_2-f_{-2}] \qquad \text{IX}$$

When this formula was used with $f(x)=\sin x$, the average absolute error over the range $41\frac{1}{2}^\circ$ to $48\frac{1}{2}^\circ$, the average absolute error was .000055. Thus, for continuous derivatives and covering unit range, we have

No. of points	Average absolute error
3	.001227
4	.000735
5	.000055

Thus the formulae involving an odd number of points seem more accurate, per point involved. If, however, one does not require continuity, the Bessel formula, involving 4 points, is almost as accurate as this 5 point formula.

Comparison of Speed of Various Formulae

The time required for running the present Stormesh program on the Illiac is roughly 15-18 seconds per sector. An 80 sector run requires 20-24 minutes. Longer runs will certainly be desired in the future, and the speed of operation of the program is important. The IBM 704 computer is considerably faster than the Illiac, by roughly a factor of 3 for fixed-point operations, and perhaps about the same speed if floating-point operations are used (of course, the use of floating-point operations should reduce somewhat the number of operations for a 704 Stormesh program, since no scaling should be needed). A further factor in the speed comparison is the availability of automatic cycle-counting and address-modification instructions for the 704. Since all practical computational programs, as distinct from data-processing, contain cycles of operations on sequences of numbers, and since cycle-counting and address modification often require

a substantial percentage of the time for these cycles, and average speed improvement of perhaps 50% would be obtained. Thus, if one translated the present Scapocyl or Stormesh program into a 704 program, using fixed point operations, the 704 program might be faster by a factor of 4 or 5. If floating-point is used, the program would not be much faster, but scaling difficulties would be removed.

Another advantage obtained with the 704 is the size of the high-speed memory. Two core memory banks, containing a total of 8192 registers, each containing 36 bits, are expected for the 704, as contrasted with Illiac's 1024 high-speed registers of 40 bits each. With this size memory, it would not be necessary to store part of a potential mesh on the magnetic drum, or store 2 mesh values in 1 register, as is done in the present Illiac program. This both speeds and simplifies the program.

However, this greatly increased storage capacity also provides for a different approach to the computational problem. One can now consider, for example, storing the field $\vec{B}$ instead of the magnetostatic potential. This requires storage of 3 times the number of items, and if only 1 component is stored per register, the same mesh sizes as are now used (in η and ξ) would require 6 times the present storage, with about $10\frac{1}{2}$ decimal digits per number. Since at present a maximum of about 700 Illiac registers are used for mesh storage, the same mesh size would require 4200 registers of the 704 or $\frac{1}{2}$ the storage. However, if it is also desired to store the field on a finer mesh, the high-speed memory would be filled with the fine mesh if the present intervals were halved in either direction. However, the drum storage associated with the 704 can provide this greater storage for finer meshes by storage of part of the mesh on the drum. Although the access time for single words stored on the magnetic drums is very long, if one uses the drum only for transfers of large blocks of numbers, the average access time is quite short. For example, the average time required to transfer 1000

words from drum to core memory should be about 8 1/3 milliseconds for the proper point on the drum to reach the read-record heads, and then about 0.2 seconds for the transfer, or an average of about 204 m.sec/number. If this does not have to be done often during a computation, the lengthening of the program time could be kept quite small. The Stormesh program is particularly susceptible to this treatment because the problem has been reduced to one of 2 coordinates, and motion seems to progress monotonically in the ξ direction, so that one could easily split the mesh into a number of sections, storing all but one of these on the drum, and writing the old section back on the drum, and bringing the next section into the high-speed memory as required. If the present mesh size is made twice as fine, and the field $\vec{B}$ is stored, a maximum of 16,800 registers would be required. It would seem reasonable that the mesh be broken up into 3 parts of about 5600 registers each*, assuming 1800 registers are sufficient for the new Stormesh program. Then 3 times each sector, 5600 words would have to be written on the drum, and 5600 read from it, requiring about 1.1 seconds each time, or 3.3 seconds per sector. If floating-binary arithmetic is used, the computational time/sector may be on the order of 8 seconds (since not all operations will be floating point and the cyle counters will also help speed the program). If fixed-point operations are used, the computation time might be reduced to 3 or 4 seconds per sector; in this case the use of the drum would double the total time required to 6 to 8 seconds/sector.

If $\vec{B}$ is to be stored on a mesh, the proper values of $\vec{B}$ must be first obtained. One method of doing this is to use the present Scapocyl program, and add to it a procedure for obtaining the necessary derivaties of Ω or Ω/n. Since this need be done only once for a particular problem, much more accurate interpolation formulas can be used. Since there are errors in Ω, however, these errors may be propagated and become larger when the components of $\vec{B}$ are calcu-

- -

* with the use of an interpolation scheme involving a number of points, some overlap between these portions of the mesh would be needed.

lated. An alternate method is to solve the potential problem for each of the 3 components of B, since all must satisfy a form of Laplace's equation. Here the correct boundary conditions must be assigned for these components of $\vec{B}$ and 3 times the labor of the present Scapocyl would be required. However, this system has the advantage that the values of $\vec{B}$ on the median plane and the pole faces can be specified; those on the median plane are directly connected with the equations of motion, while those at the pole pieces can be made in conformity with practical design.

A more direct method is to use the Powell series expansion for $\vec{B}$ to obtain values of its components at the mesh intervals. The coefficients are established by the desired nature of the function on the median plane. The field $\vec{B}$ can be calculated with great accuracy by using a sufficiently large number of terms in the expansion; since these calculations need only be done once, it does not matter if these computations are quite lengthy.

In all Stormesh-type programs with $\vec{B}$ stored, the analytic nature of $\vec{B}$ as a function of ξ and n is established by the interpolation formula used. If $\vec{B}$ is stored instead of Ω, continuity of B is given by continuity of the interpolation formula. This continuity is concomitant with accuracy of the formula, and therefore continuity of $\vec{B}$, which was obtained only with considerable loss of accuracy when Ω was stored, is obtained (naturally) if $\vec{B}$ is stored.

However, the condition that $\nabla \cdot \vec{B} = 0$ will <u>not</u> in general be satisfied by the function $\vec{B}$ as given by the interpolation formula and the mesh values. Then no vector potential $\vec{A}$ exists such that $\vec{B} = \nabla \times \vec{A}$, and this implies that the equations of motion will not be exactly Hamiltonian, as they should be. No simple technique for modifying $\vec{B}$ so that $\nabla \cdot \vec{B} = 0$ <u>and</u> so that $\vec{B}$ is continuous at the mesh points and periodic in ξ across the mesh has been found. If some way of modifying the $\vec{B}$ component values stored on the mesh can be derived which

makes $\nabla \cdot \vec{B} = 0$, perhaps substantial time could be used to accomplish this during the program which prepares the mesh values. If instead, a method is devised which is used during the operation of the Stormesh program, the operation time for this program may be extended substantially.

A recent communication from L. J. Laslett* indicates that further tests using the Stormesh overwrite program (with Bessel's interpolation and interpolation-differentiation formulae) have checked previous similar runs made with the original program. For motion in the median plane, the results with this overwrite are in better agreement with the Ridge-Runner results. However, at least one case where y motion exists has blown up with both Stormesh programs, but the motion is stable as given by the Feckless Five program. It still is not know whether this is due to the nature of the Stormesh program (and the fact that $\nabla \cdot \vec{B} \neq 0$ for its fields) or the errors of the Feckless Five program.

Among possible improvements and changes which might be considered in rewriting the Stormesh and Scapocyl programs for the 704 computer are the following:

1. Two-dimensional interpolation formulae can be devised which incorporate in them, to some extent at least, the requirement that the functions which they determine be solutions of a particular partial differential equation, as Laplace's equation. This simply gives certain conditions on the coefficients of the polynomial used. To satisfy these conditions and continuity requirements (as well as periodicity in ξ) is difficult. However, continuity, per se, may not be as important as hitherto regarded. One might be able to improve the situation in regard to $\nabla \cdot \vec{B}$ at the expense of continuity of $\vec{B}$ itself at the mesh points, but the above-mentioned results indicate that this may not be too critical at present.

- - - - - - - - - - - - - - - - -

* Letter of 23 August, 1956.

2. To obtain greater speed and accuracy in finding values of Ω or B, it might be possible to store the coefficients of an approximation function for these quantities in the η direction, instead of mesh values. For the potential function Ω near a pole edge, variation of Ω with η was found to be quite rapid, too much so to fit well with a simple polynomial. This fact makes somewhat dubious the use of Ω/η instead of Ω in hopes of obtaining better results for interpolation and differentiation, except perhaps very near the median plane. For if f(x) can not be represented very accurately by a polynomial, f(x)/x can not be either. In view of the additional complications introduced by the storage of Ω/η, such as needing an interpolation formula in addition to one for interpolation-differentiation, one might simply try to use the optimum differentiation-interpolation on the stored values of Ω, rather than a formula which fits the function itself well.

For functions which vary rapidly at a certain point, curve-fitting by a quotient of polynomials is often better than by the usual power series. One reason for this is that the ratio P(x)/Q(x), where P and Q are polynomials in x, can be fashioned with singularities where needed. If such a rational polynomial expression were to be used, by storage of the coefficients of these polynomials, the storage requirements might be no more than those presently required for the mesh values, and the problem of continuity would be satisfied automatically. The time required to calculate such a quotient is only roughly one division time more than that required for a polynomial of degree equal to the sum of the degrees of P and Q, and this time should be less than that presently required for the interpolation formula. If this function should fit the potential well enough, its derivitive, which can be obtained analytically but which would require more computation time, could be used, and again the requirement of continuity is met.

3. The present Stormesh program output consists of a table of values of x, p_x, y and p_y for a point in each sector. These results are then plotted on phase diagrams, and from these plots σ_x and σ_y are determined. Since these σ's are important factors in the selection of suitable designs, their values are often sufficient to determine that some changes in the machine parameters will be necessary. The labor involved in these plots and measurements is considerable and the source of delays in proceeding to new trial designs. This suggests that some consideration be given to automatizing these procedures and incorporating them into any new Stormesh program for the 704.

The scope output of the 704 suggests itself for automatic plotting of these phase diagrams. Since σ_x and σ_y are determined by the ratios of the numbers of times these points move around the closed phase diagrams of the number of sectors traversed, one could label each point as it is printed on the scope, by printing a small number or symbol near each point as it is printed. However, one might carry this process a step further, and by transferring to a coordinate system whose origin is at the fixed point for the phase diagram, compute the points in polar coordinates about this center. By counting the change in θ in revolutions, the computer can calculate σ_x and σ_y without necessarily plotting the phase diagram at all, although one would probably want to do this also, as a check. Also, this method presupposes a closed phase diagram, and one might use the phase diagram to verify this.

One can store both the Scapocyl and Stormesh programs on the drums or tape units, so that both are quickly available for further runs, with desired changes in parameters made in some standard way. Whether the output of the Stormesh program could be used to modify the parameters for the Scapocyl, and restart it automatically, is a possibility which might be considered as the ultimate goal in this direction.

TABLE I

ERRORS ($\times 10^{-6}$)

$f(x) = \sin x$ $\quad h = 7^{\circ}$

$f'(x) = \cos x$ $\quad u = 0,1(.1)$

	Correct	Four-point Formulas				Three-point	Five-point
X	$f'(x)$	II	III	IV	VI*	VII	
38.0°	.788011	-1958	-1958	-488	-95		
38.7	.780430	-938	-1274	-435	-82		
39.4	.772734	-139	-735	-380	-65		
40.1	.764921	441	-341	-323	-43		
40.8	.756995	800	-94	-267	-19		
41.5	.748956	937	6	-212	6	-460	6
42.2	.740805	850	-44	-160	31	-938	74
42.9	.732543	539	-243	-110	55	-1305	96
43.6	.724172	2	-594	-66	76	-1562	83
44.3	.715693	-762	-1097	-28	94	-1712	46
45.0	.707107	-1755	-1755	2	107 -110	-1755	-1
45.7	.698415	-852	-1149	25	-96	-1692	-48
46.4	.689620	-143	-670	36	-77	-1525	-84
47.1	.680721	374	-317	37	-54	-1255	-95
47.8	.671721	696	-94	25	-28	-884	-72
48.5	.662620	822	0	0	0	-411	0
49.2	.653421	751	-39	-42	27		
49.9	.644124	481	-210	-99	54		
50.6	.634731	12	-536	-175	78		
51.3	.625243	-659	-955	-270	98		
52.0	.615661	-1532	-1532	-386	113		
Av. Absolute Error		735	650	170	62	1227	55

*Derivative discontinuous at mesh points.

APPENDIX I

Stormesh Overwrite for Bessel's Formula (discontinuous derivatives)

	00129K	Directive		
129	L5 6F	f_2	$f = f_o + a_1u + a_2u^2 + a_3u^3$	
	L0 3F	$f_2 - f_{-1}$	$hf^1 = a_1 + 2a_2u + 3a_3u^2$	
	40 2F	$f_2 - f_{-1} \rightarrow 2F$		
	50993F	$1/3 \rightarrow Q$		3F. f_{-1}
	7J 2F	$1/3(f_2-f_{-1})$		4F. f_o
	L4 4F	$1/3(f_2-f_{-1}) + f_o$		5F. f_1
	L0 5F	$1/3(f_2-f_{-1}) + f_o-f_1$		6F. f_2
	10 1F	$1/6(f_2-f_{-1}) + 1/2(f_o-f_1)=a_3$		8F. u
	40 2F	$a_3 \rightarrow 2F$		
	L5 5F	f_1		
	L4 3F	f_1+f_{-1}		
	10 1F	$1/2(f_1+f_{-1})$		
135	L0 4F	$1/2(f_1+f_{-1})-f_o = a_2$		
	40 1F	$a_2 \rightarrow 1F$		
	L5 5F	f_1		
	L0 3F	f_1-f_{-1}		
	10 1F	$1/2(f_1-f_{-1})$		
	L0 2F	$1/2(f_1-f_{-1})-a_3 = f_1-1/3f_{-1} -1/2f_o - 1/6f_2 = a_1$		
	40 0F	$a_1 \rightarrow 0F$		
	50 2F			
	7J 8F	a_3u		
	40 2F	$a_3u \rightarrow 2F$		
140	L4 1F	$a_3u + a_2$		
	40 1F	$a_3u + a_2 \rightarrow 1F$		
	00 1F	$2a_3u + 2a_2$		
	L4 2F	$3a_3u + 2a_2$		
	40 2F	$3a_3u + 2a_2 \rightarrow 2F$		

	50 8F	
	7J 2F	$3a_3u^2 + 2a_2u$
	L4 0F	$3a_3u^2 + 2a_2u + a_1 = hf^1$
	50 1F	$a_3u + a_2 \rightarrow Q$
	40(12F)	store hf^1 in 12F (later 13F, 14F, 15F)
145	7J 8F	$a_3u^2 + a_2u$
	L4 0F	$a_3u^2 + a_2u + a_1$
	40 2F	$a_3u^2 + a_2u + a_1 \rightarrow 2F$
	50 8F	
147	26 991F	out to patch
	40 (16F)	store f in 16F (later 17F, 18F, 19F)
	00991K	Directive
991	7J 2F	$a_3u^3 + a_2u^2 + a_1u$
	L4 4F	$a_3u^3 + a_2u^2 + a_1u + a_o = f$
	50 2s3	load Q register
	22 147F	return to right side of 147
993	00F 00333	1/3
	333 333 333J	
	00 11K	Directive
	K6 991F	constant for stopping loop
	40 20F	
	0016K	Directive
16	F0 1L	check for mesh size
	36 19F	
	26 12F	
	00 678F	
	26 12F	
	00( )F	
	24 999 N	starting instructions
	24 500 N	

Patch