A NONLINEAR LEAST SQUARES PROGRAM BASED ON DIFFERENTIAL EQUATIONS, MULTI (RUNGE), FOR MICROCOMPUTERS

KIYOSHI YAMAOKA AND TERUMICHI NAKAGAWA

Faculty of Pharmaceutical Sciences, Kyoto University, Sakyo-ku, Kyoto, 606, Japan

(Received March 11, 1983)

A nonlinear least squares program based on first-order simultaneous differential equations, MULTI (RUNGE), for personal computers was developed. Four algorithms, *i.e.* the classical Gauss-Newton method, the damping Gauss-Newton method, the modified Marquardt method and the simplex method can be used for the nonlinear curve fitting in MULTI (RUNGE). The differential equations and the initial conditions which are voluntarily defined by the user are numerically solved by the Runge-Kutta-Gill method and the numerically integrated solutions are fitted to the observed data points by the iterative least squares algorithms. MULTI(RUNGE) can be executed on many personal computers without any modification, because this program is written in the Microsoft minimum BASIC commands alone. The dimensions of differential equations, the number of parameters to evaluate and the number of experimental data points are restricted only by the available memory in computers and by the computing time.

Keywords — pharmacokinetics; MULTI (RUNGE); AIC; curve fitting; micro-computer; least square; differential equation

INTRODUCTION

In the previous report,¹⁾ a nonlinear curve fitting program (MULTI) which simultaneously fits the explicit theoretical equations to the experimental data points was introduced for personal computers. However, the increase in the dimensions of the simultaneous ordinary differential equations which are the mathematical basis in the compartment models and the physiological models²⁾ makes it difficult to obtain the explicit analytical solutions. When the differential equations express a nonlinear pharmacokinetic process such as the Michaelis-Menten elimination or the Langmuir distribution, the analytical solution is usually impossible excepting the most simple nonlinear models. MULTI (RUNGE), therefore, was developed for personal computers. MULTI (RUNGE) numerically solves the simultaneous ordinary differential equations by means of the Runge-Kutta-Gill method, and the numerically integrated solutions are fitted to the observed time course data.

A. Hardware

MULTI (RUNGE) can be executed without any modification on the personal computers on which the Microsoft BASIC is available. This program occupies about 10K bytes for its own memory area. The dimensions of differential equations, the number of parameters to estimate and the number of experimental data points are restricted only by the available memory in computers and by computing time. Since the execution of MULTI (RUNGE) needs much time compared with that of MULTI, the use of personal computer with 16-bit microprocessor or the BASIC compiler is desirable. The output device is limited to the cathode ray tube (CRT). The command, PRINT, must be replaced by the command to printer such as LPRINT or PRINT #1, when the output to printer is required.

B. Nonlinear Least Squares Algorithms

Four algorithms, *i.e.* the classical Gauss-Newton method, the damping Gauss-Newton

MULTI < RUNGE >

```

10 PRINT"*****"
15 PRINT"* MULTI-LINES FITTINGS SEPT 21, 1982 *"
16 PRINT"*      BY RUNGE-KUTTA-GILL METHOD      *"
17 PRINT"*****"
20 PRINT:PRINT"DEFINE DIFFERENTIAL EQUATIONS FROM 1000 THROUGH 1500."
25 PRINT"DEFINE INITIAL CONDITIONS FROM 1600 THROUGH 1700."
30 PRINT"C(I) AND T ARE DEPENDENT AND INDEPENDENT VARIABLES, RESPECTIVELY."
31 PRINT"DC(I) IS DIFFERENTIAL OF C(I).":SQ=SQR(0.5)
32 PRINT"P(1), P(2),... ARE PARAMETERS TO FIT."
33 PRINT"C0(1),C0(2),... ARE INITIAL CONDITIONS."
36 PRINT:PRINT"      GOTO 840 IF DIVERGED."
37 DIM ME$(3):ME$(0)="GAUSS-NEWTON":ME$(1)="DAMPING GAUSS-NEWTON"
38 ME$(2)="MODIFIED MARQUARDT":ME$(3)="SIMPLEX"
40 PRINT:FOR I=0 TO 3:PRINT"(<I>)" :ME$(I):" METHOD":NEXT I
45 PRINT:INPUT"WHICH ALGORITHM DO YOU SELECT":AL
47 IF (AL<>0)*(AL<>1)*(AL<>2)*(AL<>3) THEN 45
49 PRINT:PRINT"* I BELIEVE YOU HAVE DEFINED EQUATIONS *"
50 INPUT"SUBJECT NAME":N$:INPUT"NUMBER OF TIME COURSES":LN
52 INPUT"NUMBER OF PARAMETERS":M:DIM NL(LN)
53 INPUT"WEIGHT OF DATA(0,1,2)":IW:IF (IW<>0)*(IW<>1)*(IW<>2) THEN 53
55 NM=0:FOR I=1 TO LN
56 PRINT"NUMBER OF POINTS(<I>)":INPUT NL(I):IF NM<NL(I) THEN NM=NL(I)
57 NEXT I:DIM XT(NM),YT(NM),ZT(NM):N=0:FOR I=1 TO LN:N=N+NL(I):NEXT I
60 DIM TX(N),CY(N),A(M,M+1),P(M),X0(M),X1(M),LS(M,M+1)
65 NL(0)=0:BS=0:FOR J=1 TO LN:BS=BS+NL(J-1):PRINT:FOR I=1 TO NL(J)
67 PRINT"(">I>)" :CP":J":<I>)" :INPUT TX(BS+I),CY(BS+I):NEXT I:NEXT J
70 PRINT:INPUT"NUMBER OF DIFF. EQUATIONS":ND
71 PRINT:INPUT"CALCULATING INTERVAL(0.1-0.01)":DT
72 TL=0:FOR I=1 TO N:IF TL<TX(I) THEN TL=TX(I)
73 NEXT I:NN=INT(TL/DT)+2:DIM DC(ND),C0(ND),C(ND),Q(ND),R(ND),K(ND),TT(NN)
74 DIM CC(NN,ND),CT(NN,LN)
76 IF AL=3 THEN 3000
77 PC=0.0001:CF=100:IF FP<>1 THEN DIM CS(N,M):FP=1
78 PRINT:INPUT"DP FOR JACOBIAN (0.1-0.0001)":DP:PRINT:FOR I=1 TO M
80 PRINT"INITIAL P(<I>)":INPUT P(I):A(I,0)=P(I):NEXT I:GOSUB 4000:S1=SS
140 PRINT"INITIAL SS=":SS:FOR KK=1 TO 100:GOSUB 7000:GOSUB 7400:GOSUB 6000:JJ=0
490 JJ=JJ+1:IF JJ>25 THEN PRINT"OUT OF DAMPING !":GOTO 730
500 FOR I=1 TO M:P(I)=A(I,0)+A(I,M+1):NEXT I:GOSUB 4000
510 DS=ABS(S1-SS):IF (AL<>2)+(SS=0) THEN 590
515 REM*** FLETCHER MODIFICATION ***
520 PW=0:FOR I=1 TO M:PW=PW+X0(I)*A(I,M+1)+CF*A(I,M+1)*A(I,M+1):NEXT I
530 IF DS>0.75*PW THEN CF=CF/2:GOTO 590
540 IF DS<0.25*PW THEN CF=5*CF
590 IF DS<=PC*S1 THEN 730:REM*** CHECK OF CONVERGENCE ***
595 REM*** DAMPING ***
600 IF (AL=1)*(SS>S1) THEN FOR I=1 TO M:A(I,M+1)=0.5*A(I,M+1):NEXT I:GOTO 490
630 FOR I=1 TO M:A(I,0)=P(I):NEXT I:S1=SS:PRINT:PRINT"LOOP=":KK
640 IF AL=1 THEN PRINT"DAMP=":JJ
650 IF AL=2 THEN PRINT"FACTOR=":CF
660 FOR I=1 TO M:PRINT"P(<I>)":P(I):NEXT I:PRINT"SS=":SS:NEXT KK
730 REM*** LOOP EXIT ***
731 PRINT:PRINT"*>N$>*" BY ">ME$(AL)>*" METHOD":PRINT"WEIGHT=1/CP^(<I>)"
733 PRINT"LOOP=":KK:IF (AL<>3)*(N>M) THEN 8000
734 IF SS=0 THEN PRINT"AIC=-INFINITE":GOTO 740
735 PRINT"AIC=">N$>LOG(SS)+2*M
740 IF AL=3 THEN PRINT"ALPHA=">AA> BETA=">BB> GAMMA=">CC>PRINT
742 IF AL<>3 THEN PRINT"DP=">DP>
750 FOR I=1 TO M:PRINT"FINAL P(<I>)":P(I):
760 IF (AL<>3)*(X0(I)>0)*(N>M) THEN PRINT" S.D.=">SQR(X0(I)*SS/(N-M)):
810 PRINT:NEXT I:PRINT"FINAL SS=":SS:BS=0:PRINT:GOSUB 9200
815 FOR J=1 TO LN:BS=BS+NL(J-1):PRINT:FOR I=1 TO NL(J):T=TX(BS+I):XT(I)=T
820 GOSUB 9300:ZT(I)=CP:YT(I)=CY(BS+I):NEXT I:NS=NL(J)
827 X$="T"+STR$(J):Y$="CP"+STR$(J):GG=1:GOSUB 10000:GOSUB 9200:FOR I=1 TO NL(J)

```

```

830 PRINT "T"; J; "="; XT(I); "      CP"; J; "="; ZT(I); "  ("; CY(BS+I); ")"
831 NEXT I:PRINT:GOSUB 9200:NEXT J
835 REM-- INSERT PROGRAM LINES FROM 832 THROUGH 839 IF NEEDED. --
840 PRINT:PRINT"WHICH ALGORITHM DO YOU SELECT"
850 INPUT "(0,1,2,3 OR -1)";AL:IF AL<0 THEN END
855 IF (AL<>0)*(AL<>1)*(AL<>2)*(AL<>3) THEN 850
857 PRINT:PRINT"**** ";ME$(AL);" METHOD ****"
860 GOTO 76:REM** RUNGE-KUTTA-GILL METHOD **
900 GOSUB 1600:FOR I1=1 TO ND:Q(I1)=0:C(I1)=C0(I1):NEXT I1
905 T=0:FOR J1=1 TO NN:TT(J1)=T:FOR I1=1 TO ND:CC(J1,I1)=C(I1):NEXT I1
910 GOSUB 1000:FOR I1=1 TO ND:K(I1)=DT*DC(I1):R(I1)=(K(I1)-2*Q(I1))/2
915 C(I1)=C(I1)+R(I1):Q(I1)=Q(I1)+3*R(I1)-K(I1)/2:NEXT I1
920 GOSUB 1000:FOR I1=1 TO ND:K(I1)=DT*DC(I1):R(I1)=(1-SQ)*(K(I1)-Q(I1))
925 C(I1)=C(I1)+R(I1):Q(I1)=Q(I1)+3*R(I1)-(1-SQ)*K(I1):NEXT I1
930 GOSUB 1000:FOR I1=1 TO ND:K(I1)=DT*DC(I1):R(I1)=(1+SQ)*(K(I1)-Q(I1))
935 C(I1)=C(I1)+R(I1):Q(I1)=Q(I1)+3*R(I1)-(1+SQ)*K(I1):NEXT I1
940 GOSUB 1000:FOR I1=1 TO ND:K(I1)=DT*DC(I1):R(I1)=(K(I1)-2*Q(I1))/6
945 C(I1)=C(I1)+R(I1):Q(I1)=Q(I1)+3*R(I1)-K(I1)/2:NEXT I1
950 T=T+DT:NEXT J1:RETURN
999 REM** DEFINE DIFFERENTIAL EQUATIONS BELOW **
1000 DC(1)=-P(1)*C(1)
1500 RETURN
1599 REM** DEFINE INITIAL CONDITIONS BELOW **
1600 C0(1)=P(2)
1700 RETURN
2999 REM*** SIMPLEX METHOD ***
3000 AA=1:BB=0.5:CC=2:SG=1E+10:PC=0.00001
3025 PRINT:FOR I=1 TO M:PRINT"INITIAL P(";I;")":INPUT A(I,1):NEXT I
3030 FOR J=2 TO M+1:FOR I=1 TO M:A(I,J)=2*RND(1)*A(I,1)+0.01*(RND(1)-0.5)
3035 NEXT I:NEXT J:KK=0
3040 FOR K=1 TO M+1:FOR I=1 TO M:P(I)=A(I,K):NEXT I:GOSUB 4000:A(0,K)=SS:NEXT K
3070 KK=KK+1:IF KK>500 THEN 730
3075 PRINT:PRINT"LOOP=";KK:FOR I=1 TO M+1:PRINT"SS";I;"=";A(0,I):NEXT I
3077 GOTO 5000
3080 SR=0:SL=1E+10:FOR J=1 TO M+1:IF SR<A(0,J) THEN JH=J:SR=A(0,J)
3090 IF SL>A(0,J) THEN JL=J:SL=A(0,J)
3100 NEXT J:SR=0:FOR J=1 TO M+1:IF (J<>JH)*(SR<A(0,J)) THEN JS=J:SR=A(0,J)
3110 NEXT J:FOR I=1 TO M:X0(I)=0:FOR J=1 TO M+1
3115 IF J<>JH THEN X0(I)=X0(I)+A(I,J)
3120 NEXT J:X0(I)=X0(I)/M:NEXT I:FOR I=1 TO M:A(I,0)=(1+AA)*X0(I)-AA*A(I,JH)
3130 P(I)=A(I,0):NEXT I:GOSUB 4000:SR=SS:IF SR<A(0,JS) THEN 3300
3160 IF SR<A(0,JH) THEN FOR I=1 TO M:A(I,JH)=A(I,0):NEXT I:A(0,JH)=SR
3170 FOR I=1 TO M:A(I,0)=BB*A(I,JH)+(1-BB)*X0(I)
3180 P(I)=A(I,0):NEXT I:GOSUB 4000:SR=SS
3190 IF SR<A(0,JH) THEN FOR I=1 TO M:A(I,JH)=A(I,0):NEXT I:A(0,JH)=SR:GOTO 3070
3200 FOR K=1 TO M+1:FOR I=1 TO M:A(I,K)=(A(I,K)+A(I,JL))/2:P(I)=A(I,K):NEXT I
3210 GOSUB 4000:A(0,K)=SS:NEXT K:GOTO 3070
3300 IF SR<A(0,JL) THEN 3500
3320 FOR I=1 TO M:A(I,JH)=A(I,0):NEXT I:A(0,JH)=SR:GOTO 3070
3500 FOR I=1 TO M:X1(I)=CC*A(I,0)+(1-CC)*X0(I)
3505 P(I)=X1(I):NEXT I:GOSUB 4000:SL=SS
3510 IF SL<SR THEN FOR I=1 TO M:A(I,JH)=X1(I):NEXT I:A(0,JH)=SL:GOTO 3070
3520 GOTO 3320
3999 REM*** CALCULATION OF SS ***
4000 GOSUB 900:SS=0:BS=0:FOR J=1 TO LN:BS=BS+NL(J-1):FOR I=1 TO NL(J)
4020 T=TX(BS+I):GOSUB 9300:TE=(CY(BS+I)-CP)*(CY(BS+I)-CP)
4022 IF IW=0 THEN SS=SS+TE:GOTO 4030
4024 IF IW=1 THEN SS=SS+TE/CY(BS+I):GOTO 4030
4026 IF IW=2 THEN SS=SS+TE/CY(BS+I)/CY(BS+I)
4030 NEXT I:NEXT J:RETURN
4999 REM*** CHECK OF CONVERGENCE ***
5000 SR=0:FOR I=1 TO M+1:SR=SR+A(0,I):NEXT I
5030 IF ABS(SR-SG)>PC*SG THEN SG=SR:GOTO 3080
5040 FOR I=1 TO M:P(I)=A(I,JL):NEXT I:SS=A(0,JL):GOTO 730
5999 REM** MODIFIED CHOLSKY METHOD **

```

```

6000 IF NS=1 THEN A(1,2)=A(1,2)/A(1,1):RETURN
6020 FOR JS=1 TO NS:LS(1,JS)=A(1,JS):NEXT JS
6030 FOR IS=2 TO NS:FOR JS=IS TO NS:DS=0:FOR KS=1 TO IS-1
6040 DS=DS+LS(KS,IS)*LS(KS,JS)/LS(KS,KS)
6050 NEXT KS:LS(IS,JS)=A(IS,JS)-DS:NEXT JS:NEXT IS
6060 LS(1,NS+1)=A(1,NS+1):FOR IS=2 TO NS:DS=0
6070 FOR KS=1 TO IS-1:DS=DS+LS(KS,IS)*LS(KS,NS+1)/LS(KS,KS):NEXT KS
6080 LS(IS,NS+1)=A(IS,NS+1)-DS:NEXT IS
6090 A(NS,NS+1)=LS(NS,NS+1)/LS(NS,NS)
6100 FOR IS=NS-1 TO 1 STEP -1:DS=0:FOR KS=IS+1 TO NS
6110 DS=DS+LS(IS,KS)*A(KS,NS+1):NEXT KS
6120 A(IS,NS+1)=(LS(IS,NS+1)-DS)/LS(IS,IS):NEXT IS:RETURN
6999 REM*** NORMAL EQUATION ***
7000 FOR JS=1 TO M:PT=P(JS):P(JS)=PT+DP:GOSUB 900
7010 FOR J=1 TO NN:FOR I=1 TO LN:CT(J,I)=CC(J,I):NEXT I:NEXT J
7020 P(JS)=PT-DP:GOSUB 900:NL(0)=0:BS=0:FOR J=1 TO LN:BS=BS+NL(J-1)
7030 FOR I=1 TO NL(J):T=TX(BS+I):GOSUB 9300:CS(BS+I,JS)=(CT(IP,J)-CP)/(2*DP)
7040 NEXT I:NEXT J:P(JS)=PT:NEXT JS
7310 FOR I=1 TO M:FOR J=I TO M:A(I,J)=0:FOR L=1 TO N
7320 TE=CS(L,I)*CS(L,J)
7330 IF IW=0 THEN A(I,J)=A(I,J)+TE:GOTO 7395
7340 IF IW=1 THEN A(I,J)=A(I,J)+TE/CY(L):GOTO 7395
7350 IF IW=2 THEN A(I,J)=A(I,J)+TE/CY(L)/CY(L)
7395 NEXT L:A(I,I)=A(I,I):NEXT J:NEXT I:RETURN
7400 NL(0)=0:BS=0:FOR J=1 TO LN:BS=BS+NL(J-1):FOR I=1 TO NL(J):T=TX(BS+I)
7410 GOSUB 9300:CS(BS+I,0)=CY(BS+I)-CP:NEXT I:NEXT J
7430 FOR I=1 TO M:A(I,M+1)=0:FOR J=1 TO N
7435 TE=CS(J,I)*CS(J,0)
7437 IF IW=0 THEN A(I,M+1)=A(I,M+1)+TE:GOTO 7448
7440 IF IW=1 THEN A(I,M+1)=A(I,M+1)+TE/CY(J):GOTO 7448
7445 IF IW=2 THEN A(I,M+1)=A(I,M+1)+TE/CY(J)/CY(J)
7448 NEXT J:NEXT I:NS=M
7450 REM*** MARQUARDT MODIFICATION ***
7460 IF AL=2 THEN FOR I=1 TO M:A(I,I)=A(I,I)+CF:X0(I)=A(I,M+1):NEXT I
7470 RETURN
7999 REM*** VARIANCE ***
8000 GOSUB 7000:FOR K=1 TO M:FOR I=1 TO M:A(I,M+1)=0:NEXT I:A(K,M+1)=1
8025 GOSUB 6000:X0(K)=A(K,M+1):NEXT K:GOTO 734
9199 REM ***** TEMPORARY STOP *****
9200 INPUT"DO YOU CONTINUE(Y/N)";C0$
9210 IF C0$="N" THEN END
9220 RETURN
9299 REM** T-CP SUBROUTINE **
9300 SP=1E+10:FOR I1=1 TO NN:S2=(TT(I1)-T)*(TT(I1)-T)
9310 IF SP>S2 THEN SP=S2:IP=I1:NEXT I1
9320 CP=CC(IP,J):RETURN
9999 REM***** GRAPH SUBROUTINE *****
10000 Y0=1E+10:Y9=-1E+10:X0=1E+10:X9=-1E+10
10020 FOR GI=1 TO NS
10030 IF XT(GI)>X9 THEN X9=XT(GI)
10040 IF XT(GI)<X0 THEN X0=XT(GI)
10050 IF YT(GI)>Y9 THEN Y9=YT(GI)
10060 IF YT(GI)<Y0 THEN Y0=YT(GI)
10062 IF GG<>1 THEN 10070
10064 IF ZT(GI)>Y9 THEN Y9=ZT(GI)
10066 IF ZT(GI)<Y0 THEN Y0=ZT(GI)
10070 NEXT GI
10080 IF YL$="0" THEN Y0=0
10090 IF XL$="0" THEN X0=0
10100 IF GF<>1 THEN DIM GR$(23):GF=1
10110 SP$="
10120 FOR GI=1 TO 23:GR$(GI)=SP$:NEXT GI
10130 G1=5:G2=4:G3=5:G4=6:G5=5:XM=30:YM=18:PT=1
10140 GY=(Y9-Y0)/(G2-1):GG$="┐"
10150 GP=0:IF (GY>0.1)*(Y9<1000) THEN GT$="      "+Y$:GOTO 10170

```

```

10160 GP=INT(0.4343*LOG(GY)):GT#=Y#+"(X10↑"+STR$(GP)+")"
10170 GR$(1)=GT#+RIGHT$(GR$(1),LEN(GR$(1))-LEN(GT#))
10180 PT=PT+1
10190 FOR GI=1 TO G2:G#=STR$((Y9-GY*(GI-1))/10↑GP):IF LEN(G#)<G1 THEN G#=G#+SP$
10200 G#=LEFT$(G#,G1)+GG$:GR$(PT)=G#+RIGHT$(GR$(PT),LEN(GR$(PT))-LEN(G#))
10210 PT=PT+1:IF GI=G2 THEN 10250
10220 FOR GJ=1 TO G3:GT#=LEFT$(SP$,G1)+"|"
10230 GR$(PT)=GT#+RIGHT$(GR$(PT),LEN(GR$(PT))-LEN(GT#)):PT=PT+1
10240 NEXT GJ:GG#="+" :NEXT GI
10250 G#="|":FOR GI=1 TO G4-1:G#=G#+"+||||":NEXT GI:G#=G#+"| "
10260 GT#=LEFT$(SP$,G1)+G#:GR$(PT)=GT#+RIGHT$(GR$(PT),LEN(GR$(PT))-LEN(GT#))
10270 PT=PT+1
10280 GX=(X9-X0)/(G4-1):GG#=LEFT$(SP$,G1)
10290 GP=0:IF (GX>0.1)*(X9<1000) THEN 10310
10300 GP=INT(0.4343*LOG(GX)):X#=X#+" (X10↑"+STR$(GP)+")"
10310 FOR GI=1 TO G4:G#=STR$((X0+GX*(GI-1))/10↑GP):IF LEN(G#)<4 THEN G#=G#+SP$
10320 GG#=GG#+LEFT$(G#,4):IF GI<G4 THEN GG#=GG#+LEFT$(SP$,G5-3)
10330 NEXT GI:GR$(PT)=GG#+RIGHT$(GR$(PT),LEN(GR$(PT))-LEN(GG#)):PT=PT+1
10340 GT#=LEFT$(SP$,INT(G1+(XM-LEN(X#))/2))+X#
10350 GR$(PT)=GT#+RIGHT$(GR$(PT),LEN(GR$(PT))-LEN(GT#))
10370 REM-- PLOT ROUTINE --
10380 FOR GI=1 TO NS
10400 GX=INT(XM*(XT(GI)-X0)/(X9-X0)+0.5)+G1+1
10410 GY=INT(YM*(Y9-YT(GI))/(Y9-Y0)+0.5)+1:PL$="o"
10420 GR$(GY+1)=LEFT$(GR$(GY+1),GX)+PL#+RIGHT$(GR$(GY+1),LEN(GR$(GY+1))-GX-1)
10422 IF GG<1 THEN 10430
10424 GY=INT(YM*(Y9-ZT(GI))/(Y9-Y0)+0.5)+1:PL$="*"
10426 IF MID$(GR$(GY+1),GX+1,1)="o" THEN PL$="●"
10428 GR$(GY+1)=LEFT$(GR$(GY+1),GX)+PL#+RIGHT$(GR$(GY+1),LEN(GR$(GY+1))-GX-1)
10430 NEXT GI
10440 PRINT:FOR GI=1 TO 23:PRINT GR$(GI):NEXT GI
10450 RETURN
READY.

```

FIG. 1. Program List of MULTI (RUNGE)

method,³⁾ the modified Marquardt method⁴⁾ and the simplex method⁵⁾ can be used for the nonlinear curve fitting in MULTI (RUNGE), as in MULTI. The characteristics of the four algorithms are explained in the previous report.¹⁾ The Gauss elimination method with the pivot selection which is used for solving the normal equation in MULTI is replaced by the modified Cholesky method⁶⁾ to accelerate the computing speed. MULTI (RUNGE) displays the plots of observed data point and the predicted points by the least squares. While the maximum number of equations to fit is five in MULTI, the number of differential equations is not restricted in MULTI (RUNGE). Fig. 1 is the program list of MULTI (RUNGE).

C. Definition of Differential Equations and Data Input

Before the execution of MULTI (RUNGE),

the simultaneous ordinary differential equations and the initial conditions must be defined from 1000 through 1500 and from 1600 through 1700, respectively, where C(1), C(2), ... are dependent variables, T is independent variable, DC(1), DC(2), ... are the first-order differentials of C(1), C(2), ... with respect to T, C0(1), C0(2), ... are initial conditions, and P(1), P(2), ... are parameters to evaluate. The differential equation and the initial condition which are already defined at 1000 and 1600 in Fig. 1, respectively, signify the following one-compartment model with rapid intravenous dose.

$$dC/dt = -k_e C \quad (1)$$

with the initial condition

$$C(0) = C_0 \quad (2)$$

where k_e is elimination rate constant and C_0 is the initial plasma concentration. In Fig. 1, P(1) and P(2) signify k_e and C_0 , respectively. After the run of MULTI (RUNGE), the program gives the detailed prompts for input data (i.e. the selection of nonlinear least squares algorithm, numbers of time courses, parameters to estimate and data points, and the dimensions of simultaneous differential equations, etc.).

Fig. 2 is an example of execution of MULTI (RUNGE). The adopted pharmacokinetic model is the one-compartment model with the first-order absorption (Chart 1). The reason to adopt the linear model first is to compare the result of MULTI (RUNGE) with that of MULTI. The mass-balance equation is given by

$$\begin{aligned} dC_{iv}/dt &= -k_e C_{iv} \\ dC_{po}/dt &= F(k_a/V_d)X_G - k_e C_{po} \end{aligned} \quad (3)$$

$$dX_G/dt = -k_a X_G$$

with the initial conditions,

$$C_{iv}(0) = D_{iv}/V_d, C_{po}(0) = 0 \text{ and } X_G(0) = D_{po} \quad (4)$$

where subscripts iv and po specify intravenous and oral doses, respectively, C is plasma concentration, D is dosing amount, X_G is the drug amount remaining in GI tract, F is ratio of ab-

sorbed amount, V_d is volume of distribution, k_a is absorption rate constant and k_e is elimination rate constant. The explicit solution of Eq. 1 are given as

$$\begin{aligned} C_{iv} &= (D_{iv}/V_d) e^{-k_e t} \quad (5) \\ C_{po} &= \frac{FD_{po}k_a}{V_d(k_a - k_e)} (e^{-k_e t} - e^{-k_a t}) \end{aligned}$$

In MULTI (RUNGE), the definition of the differential equations and the initial conditions are

```
999 REM** DEFINE DIFFERENTIAL
EQUATIONS BELOW**
1000 DC(1)=-P(1)*C(1)
1010 DC(2)=P(2)*P(3)/P(4)*C(3)-P(1)*C(2)
1020 DC(3)=-P(3)*C(3)
1500 RETURN
1599 REM** DEFINE INITIAL CONDI-
TIONS BELOW**
1600 C0(1)=50/P(4):C0(2)=0:C0(3)=50
1700 RETURN
```

where $P(1)=k_e$, $P(2)=F$, $P(3)=k_a$, $P(4)=V_d$, $C(1)=C_{iv}$, $C(2)=C_{po}$, $C(3)=X_G$, $D_{iv}=50$ and $D_{po}=50$.

The underlines in Fig. 2 specify the input data to

EX. OF MULTI(RUNGE)

```
999 REM** DEFINE DIFFERENTIAL EQUATIONS BELOW **
1000 DC(1)=-P(1)*C(1)
1010 DC(2)=P(2)*P(3)/P(4)*C(3)-P(1)*C(2)
1020 DC(3)=-P(3)*C(3)
1500 RETURN
1599 REM** DEFINE INITIAL CONDITIONS BELOW **
1600 C0(1)=50/P(4):C0(2)=0:C0(3)=50
1700 RETURN
```

READY.

RUN

```
*****
* MULTI-LINES FITTINGS SEPT 21, 1982 *
* BY RUNGE-KUTTA-GILL METHOD *
*****
```

```
DEFINE DIFFERENTIAL EQUATIONS FROM 1000 THROUGH 1500.
DEFINE INITIAL CONDITIONS FROM 1600 THROUGH 1700.
```

C(I) AND T ARE DEPENDENT AND INDEPENDENT VARIABLES, RESPECTIVELY.
 DC(I) IS DIFFERENTIAL OF C(I).
 P(1), P(2),... ARE PARAMETERS TO FIT.
 C0(1),C0(2),... ARE INITIAL CONDITIONS.

GOTO 840 IF DIVERGED.

(0) GAUSS-NEWTON METHOD
 (1) DAMPING GAUSS-NEWTON METHOD
 (2) MODIFIED MARQUARDT METHOD
 (3) SIMPLEX METHOD

WHICH ALGORITHM DO YOU SELECT 1

* I BELIEVE YOU HAVE DEFINED EQUATIONS *

SUBJECT NAME TEST

NUMBER OF TIME COURSES 2

NUMBER OF PARAMETERS 4

WEIGHT OF DATA(0,1,2) 0

NUMBER OF POINTS(1) 5

NUMBER OF POINTS(2) 6

T 1 (1), CP 1 (1) .5 , 77.9
 T 1 (2), CP 1 (2) 1.5 , 47.2
 T 1 (3), CP 1 (3) 2.5 , 28.7
 T 1 (4), CP 1 (4) 3.5 , 17.3
 T 1 (5), CP 1 (5) 4.5 , 10.5

T 2 (1), CP 2 (1) .5 , 18.4
 T 2 (2), CP 2 (2) 1 , 23
 T 2 (3), CP 2 (3) 1.5 , 22
 T 2 (4), CP 2 (4) 2.5 , 15.8
 T 2 (5), CP 2 (5) 3.5 , 10.1
 T 2 (6), CP 2 (6) 5 , 4.89

NUMBER OF DIFF. EQUATIONS 3

CALCULATING INTERVAL(0.1-0.01) .1

DP FOR JACOBIAN (0.1-0.0001) 1E-03

INITIAL P(1) = .2

INITIAL P(2) = 1

INITIAL P(3) = 2

INITIAL P(4) = 1

INITIAL SS= 2801.72018

LOOP= 1

DAMP= 2

P(1) = .336063645

P(2) = .646570142

P(3) = 1.24984611

P(4) = .630252105

SS= 710.317747

LOOP= 2

DAMP= 1

P(1) = .483328018

P(2) = .406579598

P(3) = 1.24561809

P(4) = .490143288

SS= 24.6911854

LOOP= 3

DAMP= 1

```

P< 1 >= .499538655
P< 2 >= .398238231
P< 3 >= 1.47093493
P< 4 >= .50077944
SS= .273212147

```

```

LOOP= 4
DAMP= 1
P< 1 >= .500585769
P< 2 >= .400053433
P< 3 >= 1.49871569
P< 4 >= .500721257
SS= .055603332

```

```

LOOP= 5
DAMP= 1
P< 1 >= .500586715
P< 2 >= .400070366
P< 3 >= 1.49876661
P< 4 >= .500720337
SS= .0552138015

```

TEST BY DAMPING GAUSS-NEWTON METHOD

WEIGHT=1/CP↑(0)

LOOP= 6

AIC=-23.8617693

DP= 1E-03

FINAL P< 1 >= .500586712 S.D.= 1.04461869E-03

FINAL P< 2 >= .400070328 S.D.= 1.1557842E-03

FINAL P< 3 >= 1.49876657 S.D.= .0111410529

FINAL P< 4 >= .500720341 S.D.= 7.13004387E-04

FINAL SS= .055214787

DO YOU CONTINUE(Y/N)Y

DO YOU CONTINUE(Y/N)Y

T 1 = .5 CP 1 = 77.9008075 (77.9)

T 1 = 1.5 CP 1 = 47.2215159 (47.2)

T 1 = 2.5 CP 1 = 28.6244987 (28.7)

T 1 = 3.5 CP 1 = 17.3514533 (17.3)

T 1 = 4.5 CP 1 = 10.5180159 (10.5)

DO YOU CONTINUE(Y/N)<Y

DO YOU CONTINUE(Y/N)<Y

T 2 = .5 CP 2 = 18.386626 (18.4)
 T 2 = 1 CP 2 = 23.005935 (23)
 T 2 = 1.5 CP 2 = 22.0194888 (22)
 T 2 = 2.5 CP 2 = 15.776986 (15.8)
 T 2 = 3.5 CP 2 = 10.1063416 (10.1)
 T 2 = 5 CP 2 = 4.08574093 (4.0)

DO YOU CONTINUE(Y/N)<Y

WHICH ALGORITHM DO YOU SELECT
 (0,1,2,3 OR -1)<-1

READY.

FIG. 2. An Example of Execution of MULTI (RUNGE)

CHART 1

MULTI (RUNGE) through the keyboard. The lines without underline are the output on CRT from MULTI (RUNGE). The number of differential equations may be greater than the

number of observed time courses. The DC(i) which has the corresponding experimental data must be defined first (that is, smaller i). For example, dC_{iv}/dt and dC_{po}/dt are first defined by DC(1) and DC(2), respectively, since C_{iv} and C_{po} have the corresponding observed time course data. AIC which is given in Fig. 2 is the Akaike's information criterion which is used for the selection of the most "proper" pharmacokinetic model from the several candidates. The applications of AIC to pharmacokinetic study are found elsewhere.^{7,8)} MULTI (RUNGE) displays the graphic plots, where O expresses the observed data point, * the predicted point by the

TABLE I. *Computing Time, Memories for Calculation and Converged Parameters by Four Nonlinear Least Squares Algorithms Using Generated Time Course Data*

Algorithms	Time (h)	Memory (K bytes)
G-N	1.53	3.56
Damping G-N	1.35	3.56
Marquardt	1.97	3.56
Simplex	6.25	3.23

Parameters	Initial	Final	True
k_e	0.2	0.5006	0.5
F	1.0	0.4000	0.4
k_a	2.0	1.499	1.5
V_d	1.0	0.5007	0.5

CHART 2

least squares and • the overlapped point. The time course data corresponding to C_{iv} and C_{po} are generated using Eq. 5. Table I shows the time and the memory needed to compute by four nonlinear least squares algorithms using CBM 3032 personal computer (Commodore). The data are weighted by unity (weight = $1/C_p(0)$) in all calculation in the present study. The pharmacokinetic parameters which converged to the same values by all algorithms in MULTI (RUNGE) coincide with the results computed by MULTI using the explicit Eq. 5. The memory for computing is about 3.5K bytes for all algorithms. The simplex method needs the longest computing time (about 6 h). The other algorithms take 1 h to 2 h.

D. Parameter Estimation of Nonlinear Models

In order to evaluate the pharmacokinetic parameters of nonlinear model, it is desirable to observe the time courses by changing dosing amount. The following is the Langmuir distribution model.⁹⁾

$$dC_p/dt = -k_e C_p / (1 + A \cdot B (B + C_p)^2) \quad (6)$$

with the initial condition

$$C_p(0) = 100, 200, \text{ and } 300 \quad (7)$$

where C_p is plasma concentration, k_e is elimination rate constant, and A and B are the parameters of the Langmuir distribution.

The initial concentration is changed from 100 to 300. The definition of differential equations and the initial conditions in MULTI (RUNGE) are given by

```

999 REM** DEFINE DIFFERENTIAL
EQUATION BELOW**
1000 DC(1) = -P(1)*C(1)/(1+P(2)*P
(3)/(P(3)+C(1))^2)
1010 DC(2) = -P(1)*C(2)/(1+P(2)*P
(3)/(P(3)+C(2))^2)
1020 DC(3) = -P(1)*C(3)/(1+P(2)*P
(3)/(P(3)+C(3))^2)
1500 RETURN

```

```

1599 REM** DEFINE INITIAL
CONDITIONS BELOW**
1600 CO(1)=100:CO(2)=200:CO(3)=300
1700 RETURN

```

where $P(1)=k_e$, $P(2)=A$ and $P(3)=B$. $C(1)$, $C(2)$ and $C(3)$ signify the plasma concentration-time courses corresponding to the initial conditions of 100, 200 and 300, respectively.

The two-compartment with the Michaelis-Menten elimination model (Chart 2) is described by the following mass balance equation.¹⁰⁾

$$\begin{aligned} dC_1/dt &= -(k_{12} + V_m/(K_m + C_1)) C_1 + K_{12} C_2 \\ (8) \\ dC_2 &= k_{21} C_1 - k_{21} C_2 \end{aligned}$$

with the initial conditions,

$$C_1(0)=100/V_1 \text{ and } 200/V_1, \text{ and } C_2(0)=0 \quad (9)$$

where C_1 and C_2 are drug concentrations in the central and the peripheral compartments, respectively, k_{12} and k_{21} are first-order partition rate constants out of and back into the central compartment, respectively, V_1 is volume of distribution of central compartment, V_m and K_m are the Michaelis-Menten elimination constants. In the above example, the dosing amount (D) is changed from 100 to 200.

The definitions of differential equations and the initial conditions are

```

999 REM** DEFINE DIFFERENTIAL
EQUATIONS BELOW**
1000 DC(1)=-(P(1)+P(2)/(P(3)+C(1)))
*C(1)+
P(1)*C(3)
1010 DC(2)=-(P(1)+P(2)/(P(3)+C(2)))
*C(2)+
P(1)*C(4)
1020 DC(3)=P(4)*C(1)-P(4)*C(3)
1040 DC(4)=P(4)*C(2)-P(4)*C(4)
1500 RETURN
1599 REM** DEFINE INITIAL

```

CONDITIONS BELOW**

```

1600 CO(1)=100/P(5):CO(2)=200/P(5)
1610 CO(3)=0:CO(4)=0
1700 RETURN

```

where $P(1)=k_{12}$, $P(2)=V_m$, $P(3)=K_m$, $P(4)=k_{21}$ and $P(5)=V_1$.

$C(1)$ and $C(2)$ signify the plasma concentrations after the administration of 100 and 200 doses, respectively. $C(3)$ and $C(4)$ are the concentrations in the peripheral compartment. The concentrations which have the corresponding observations are defined first as $DC(1)$ and $DC(2)$.

The variable DT at line 71 in Fig. 1 is the interval which is used for numerically integrating the differential equations by the Runge-Kutta-Gill method. The too small DT value extensively increases the computing time and may cause the memory-over. The optimum DT must be found by trial and error. The variable DP at line 78 is used to numerically calculate the Jacobian matrix by giving a perturbation to parameter $P(j)$. This value must also be estimated by trial and error. The variable PC at line 77 and 3000 is the criterion for convergence. If the iterative calculation falls into the vibrating state, the press on Break (or Stop) key and the substitution ($PC=1$) followed by **CONT** command terminate the computation compulsorily.

REFERENCES

- 1) K.Yamaoka, Y.Tanigawara, T.Nakagawa and T.Uno: A pharmacokinetic analysis program (MULTI) for microcomputer, *J. Pharm. Dyn.*, **4**, 879-885 (1981).
- 2) K.B.Bischoff, R.L.Dedrick and D.S.Zaharko: Preliminary model for methotriaxate pharmacokinetics, *J. Pharm. Sci.*, **50**, 149-154 (1970).
- 3) T.Nakagawa and Y.Oyanagi: *SALS user manual*, Feb. 1, 1981.
- 4) R.Fletcher: A modified Marquardt subroutine for non-linear least squares, "Tech. Report AERE R6799," U.K.Atomic Energy Authority Research Establishment, Harwell, U.K., 1971.
- 5) J.A.Nelder and R.Mead: A simplex method for function minimization, *Computer J.*, **7**, 308-313 (1965).
- 6) R.J.Melosh and R.M.Bamford: Efficient solution of lead-deflection equations, *J. Struct. Div. Proc. ASCE*, 661-676 (1969).
- 7) K.Yamaoka, T.Nakagawa and T.Uno: Application of

- Akaike's information criterion (AIC) in the evaluation of linear pharmacokinetic equations, *J. Pharmacok. Biopharm.*, **6**, 165–175 (1978).
- 8) K.Yamaoka, Y.Tanigawara, T.Nakagawa and T.Uno: Capacity-limited elimination of cefmetazole in rat, *Int. J. Pharm.*, **10**, 291–300 (1982).
- 9) J.G.Wagner: A modern view of pharmacokinetics, *J. Pharmacok. Biopharm.*, **1**, 363–401 (1973).
- 10) C.H.Lawyer, N.Gerber, R.K.Lynn and R.G.Dickinson: Application of the two-compartment open model with Michaelis-Menten elimination kinetics to valproic acid in the bile-exteriorized rat, *Res. Comm. Chem. Path. Pharmacol.*, **27**, 547–558 (1980).