[Chem. Pharm. Bull. 21(11)2549—2556(1973)] UDC 518.615.07

The Routine Fitting of Pharmacokinetic Data to Multiexponential Equation

TAMOTSU KOIZUMI, MICHIHIRO UEDA, and MASAWO KAKEMI

Faculty of Pharmaceutical Sciences, University of Toyama[1]

(Received April 16, 1973)

An iterative least square method for finding the best fit multiexponential equation to observed data is presented. A computer program is written in ALGOL, which permits the routine fitting of pharmacokinetic data. Given a set of data, number of exponential terms and initial estimates for exponential parameters, the computer program automatically proceeds to obtain a least square fit of the data by an iterative adjustment of the values of the parameters. Numerical tests on published data confirm its applicability.

As far as pharmacokinetic models are constructed with the network of first order rate processes, solution to the model, time course of the amount of species in any compartment, turns out to be a multiexponential equation, which is expressed as Eq. 1,

$$Y = A_0 + \sum_{j=1}^{n} A_j e^{-\alpha_j t} \qquad \text{Eq. 1}$$

where n is the number of exponential terms.

In fact, the fitting to multiexponential equations is often the very first step of pharmacokinetic analysis of obtained data.[2] On the other hand, mathematical basis and practical digital computer programs for model fitting have been discussed by quite a few authors so far.[3] Some of those programs,[3a,j] however, are so big in size and require so much storage area that they are inconvenient to use in a medium size computer like the one that is readily accessible by the authors of this report. Some other programs[3e,g] are intended the fitting of the data to such a specific model that they are inappropriate for general application.

The aim of the present paper is to describe a mathematical background for the routine fitting of pharmacokinetic data to multiexponential equation with a computer program suitable for a small computer with memory capacity of 4K words.[4]

Since multiexponential equation is linear with respect to coefficient, A_j, and it is nonlinear only about exponential parameter, α_j, it is reasonable to execute non-linear least square iteration[3a] exclusively with exponential parameters. By this way, size of the normal equation, and consequently necessary memory area, can be reduced to one half as well as the computation time to obtain the solution.

1) Location: *3190 Gofuku, Toyama 930, Japan.*
2) W.J. Jusko and G. Levy, *J. Pharm. Sci.*, **59**, 763 (1970).
3) *a*) M. Berman, E. Shahn and M.F. Weiss, *Biophys. J.*, **2**, 275 (1962); *b*) M. Berman, *J. Theoret. Biol.*, **4**, 229 (1963); *c*) D.W. Marquardt, *J. Soc. Indust. Appl. Math.*, **11**, 431 (1963); *d*) R. Fletcher and M.J.D. Powell, *Computer J.*, **6**, 163 (1963); *e*) R.G. Wiegand and P.G. Sanders, *J. Pharmacol. Exptl. Therap.*, **146**, 271 (1964); *f*) J. Myhill, G.P. Wadsworth and G.L. Brownell, *Biophys. J.*, **5**, 89 (1965); *g*) W. Lowenthal and B.L. Vitsky, *J. Pharm. Sci.*, **56**, 169 (1967); *h*) G.W. Stewart III, *J. Assoc. Comp. Machin.*, **14**, 72 (1967); *i*) S. Awazu, "Abstracts of Papers, First Symposium on Drug Metabolism and Action, Chiba, November," 1969, pp. 29—32; *j*) G.L. Atkins, *Biochem. Biophys. Acta*, **252**, 405 (1971); *k*) S. Awazu, M. Hanano, H. Moon, T. Fuwa, T. Iga and H. Nogami, "Absorption, Metabolism and Excretion of Drugs," ed. by K. Kakemi, Hirokawa Publishing Co., Tokyo, 1971, pp. 301—333.
4) Execution of the program reported in the present paper was carried out on OKITAC 5090C of The Computing Center, University of Toyama.

Theory

Definition of Type and Grade of Multiexponential Equation——For the purpose of convenience, multiexponential equation (Eq. 1) is classified into three types. If

$$A_0 = 0 \qquad \text{Eq. 2}$$

then the equation is type 1, which describes, for example, plasma level of unchanged drug following *i.v.* administration. If

$$\sum_{j=1}^{n} A_j = 0 \text{ and } A_0 = 0 \qquad \text{Eq. 3}$$

then it is type 2, and expresses the time course of blood level of biotransformed species. The multiexponential equation is referred to as type 3, if

$$\sum_{j=1}^{n} A_j = -A_0 \qquad \text{Eq. 4}$$

In this case Eq. 1 is also expressed as Eq. 5

$$Y = \sum_{j=1}^{n} A_j(e^{-\alpha_j t}-1) \qquad \text{Eq. 5}$$

which usually appears as an equation for the cumulative amount of unchanged or biotransformed species excreted in urine. If

$$\frac{d^iY}{dt^i} = 0 \quad (i=1, 2, \cdots, g) \quad \text{and} \quad \frac{d^{g+1}Y}{dt^{g+1}} \neq 0 \qquad \text{Eq. 6}$$

then the grade of the equation is defined as g. The grade is introduced in order to specify the feature of the function at the neighborhood of origin. For example, the equation that represents compartment C of two step consequtive first order rate process shown below is biexponential equation of type 3 and grade 1, if B=0 at t=0.

$$A \longrightarrow B \longrightarrow C$$

Non-linear Least Square Method——The weighted sum of square of difference between observed and calculated values is expressed by Eq. 7,

$$SS = \sum_{i=1}^{m} w_i(y_i - Y_i)^2 \qquad \text{Eq. 7}$$

where Y_i and y_i are, respectively, calculated and observed value at time t_i, w_i is weighting factor and m is the total number of observed values. For type 1, $\partial SS/\partial A_k=0$ and $\partial SS/\partial \alpha_k=0$ give two sets of simultaneous equations Eq. 8 and 9, respectively.

$$\sum_{j=1}^{n} A_j \sum_{i=1}^{m} w_i e^{-\alpha_k t_i} e^{-\alpha_j t_i} = \sum_{i=1}^{m} w_i y_i e^{-\alpha_k t_i} \qquad (k=1, 2, \cdots, n) \qquad \text{Eq. 8}$$

$$\sum_{j=1}^{n} A_j \sum_{i=1}^{m} w_i t_i e^{-\alpha_k t_i} e^{-\alpha_j t_i} = \sum_{i=1}^{m} w_i y_i t_i e^{-\alpha_k t_i} \qquad (k=1, 2, \cdots, n) \qquad \text{Eq. 9}$$

Least square parameters are such A_j and α_j that satisfy Eq. 8 and 9 simultaneously.
By solving Eq. 8 for A_j, it is expressed as a function of α_j.

$$A_j = f(\alpha_1, \alpha_2, \cdots, \alpha_n) \qquad \text{Eq. 10}$$

Consequently at a given time t, multiexponential equation can be considered as a function of exponential parameters, α_j, (j=1,2,,n). Using Taylor expansion and neglecting the higher order terms, Eq. 1 gives Eq. 11,

$$Y = \sum_{j=1}^{n} A_j(\alpha_1^\circ, \alpha_2^\circ, \cdots \alpha_n^\circ)e^{-\alpha_j^\circ t} + \sum_{j=1}^{n}\left(\frac{\partial Y}{\partial \alpha_j}\right)\Delta\alpha_j \qquad \text{Eq. 11}$$

where α°_j is an estimate of α_j and $\Delta\alpha_j=\alpha_j-\alpha^\circ_j$.

Eq. 11 is linear with respect to $\Delta\alpha_j$ and least square fit method with iteration[3a)] is applicable. For which normal equation is given by Eq. 12,

$$\sum_{j=1}^{n} \Delta\alpha_j \sum_{i=1}^{m} w_i\left(\frac{\partial Y}{\partial \alpha_k}\right)\left(\frac{\partial Y}{\partial \alpha_j}\right) = \sum_{i=1}^{m} w_i \Delta y_i \left(\frac{\partial Y}{\partial \alpha_k}\right) \qquad (k=1, 2, \cdots, n) \qquad \text{Eq. 12}$$

where $\Delta y_i = y_i - Y_i$ and $\partial Y/\partial\alpha_l$ (l=1,2,,n) is given by Eq. 13, which is obtained from Eq. 1 modified with Eq. 10 and differentiated with α_l.

$$\frac{\partial Y}{\partial \alpha_l} = \sum_{j=1}^{n}\left(\frac{\partial A_j}{\partial \alpha_l}\right)e^{-\alpha_j t} - A_l(\alpha_1^\circ, \alpha_2^\circ, \cdots, \alpha_n^\circ)\cdot t\cdot e^{-\alpha_l t} \qquad \text{Eq. 13}$$

where $\partial A_j/\partial \alpha_l$ ($l=1,2,,,n$) is evaluated by solving simultaneous linear equation Eq. 14, which is obtained through differentiating Eq. 8 by α_l and applying Eq. 9.

$$\sum_{j=1}^{n}\left(\frac{\partial A_j}{\partial \alpha_l}\right)\sum_{i=1}^{m} w_i e^{-\alpha_k t_i} e^{-\alpha_j t_i} = A_l \sum_{i=1}^{m} w_i t_i e^{-\alpha_k t_i} e^{-\alpha_l t_k} \qquad (k=1, 2, \cdots, n) \qquad \text{Eq. 14}$$

It should be noted that the coefficient matrix of Eq. 14 is identical with that of Eq. **8.** Therefore, two sets of these simultaneous equations are solved concurrently.

For the equations of type 2, Eq. 15 and 16 are obtained in place of Eq. 8 and 14, respectively.

$$\left\{\begin{aligned} &\sum_{j=1}^{n} A_j\left\{\sum_{i=1}^{m} w_i e^{-\alpha_k t_i} e^{-\alpha_j t_i} - \sum_{i=1}^{m} w_i e^{-\alpha_n t_i} e^{-\alpha_j t_i}\right\} \\ &\qquad = \sum_{i=1}^{m} y_i w_i e^{-\alpha_k t_i} - \sum_{i=1}^{m} y_i w_i e^{-\alpha_n t_i} \qquad (k=1, 2, \cdots, n-1) \\ &\sum_{j=1}^{n} A_j = 0 \end{aligned}\right. \qquad \text{Eq. 15}$$

$$\left\{\begin{aligned} &\sum_{j=1}^{n}\left(\frac{\partial A_j}{\partial \alpha_l}\right)\left\{\sum_{i=1}^{m} w_i e^{-\alpha_k t_i} e^{-\alpha_j t_i} - \sum_{i=1}^{m} w_i e^{-\alpha_n t_i} e^{-\alpha_j t_i}\right\} \\ &\qquad = A_l\left\{\sum_{i=1}^{m} w_i t_i e^{-\alpha_k t_i} e^{-\alpha_l t_i} - \sum_{i=1}^{m} w_i t_i e^{-\alpha_n t_i} e^{-\alpha_l t_i}\right\} \\ &\qquad\qquad (k=1, 2, \cdots, n-1), \ (l=1, 2, \cdots, n) \\ &\sum_{j=1}^{n}\left(\frac{\partial A_j}{\partial \alpha_l}\right) = 0 \qquad (l=1, 2, \cdots, n) \end{aligned}\right. \qquad \text{Eq. 16}$$

For the equations of type 3, Eq. 17 and 18 should be used in stead of Eq. 8 and 14, respectively.

$$\left\{\begin{aligned} &\sum_{j=1}^{n} A_j\left\{\sum_{i=1}^{m} w_i(e^{-\alpha_k t_i}-1)(e^{-\alpha_j t_i}-1) + \sum_{h=n-g+1}^{n}\left(\frac{\partial A_h}{\partial A_k}\right)\sum_{i=1}^{m} w_i(e^{-\alpha_h t_i}-1)(e^{-\alpha_j t_i}-1)\right\} \\ &\qquad = \sum_{i=1}^{m} y_i w_i(e^{-\alpha_k t_i}-1) + \sum_{h=n-g+1}^{n}\left(\frac{\partial A_h}{\partial A_k}\right)\sum_{i=1}^{m} y_i w_i(e^{-\alpha_h t_i}-1) \\ &\qquad\qquad (k=1, 2, \cdots, n-g) \\ &\sum_{j=1}^{n} \alpha_j^k A_j = 0 \qquad (k=1, 2, \cdots, g) \end{aligned}\right. \qquad \text{Eq. 17}$$

$$\left\{\begin{aligned} &\sum_{j=1}^{n}\left(\frac{\partial A_j}{\partial \alpha_l}\right)\left\{\sum_{i=1}^{m} w_i(e^{-\alpha_k t_i}-1)(e^{-\alpha_j t_i}-1) + \sum_{h=n-g+1}^{n}\left(\frac{\partial A_h}{\partial A_k}\right)\sum_{i=1}^{m} w_i(e^{-\alpha_h t_i}-1)(e^{-\alpha_j t_i}-1)\right\} \\ &\qquad = A_l\left\{\sum_{i=1}^{m} w_i t_i(e^{-\alpha_k t_i}-1)e^{-\alpha_l t_i} + \sum_{h=n-g+1}^{n}\left(\frac{\partial A_h}{\partial A_k}\right)\sum_{i=1}^{m} w_i t_i(e^{-\alpha_h t_i}-1)e^{-\alpha_l t_i}\right\} \\ &\qquad\qquad (k=1, 2, \cdots, n-g) \\ &\sum_{j=1}^{n} \alpha_j^k\left(\frac{\partial A_j}{\partial \alpha_l}\right) = -k\alpha_l^{k-1} A_l \qquad (k=1, 2, \cdots, g) \end{aligned}\right. \qquad \text{Eq. 18}$$

where g is the grade of the equation and $\frac{\partial \mathrm{A}_h}{\partial \mathrm{A}_k}$ ($h=n-g+1, \cdots, n$ and $k=1, 2, \cdots, n-g$) is evaluated through solving simultaneous linear equation Eq. 19.

$$\sum_{h=n-g+1}^{n} \alpha_h^l\left(\frac{\partial A_h}{\partial A_k}\right) = -\alpha_k^l \qquad (l=1, 2, \cdots, g), \ (k=1, 2, \cdots, n-g) \qquad \text{Eq. 19}$$

Computer Program——On preparing a practical computer program (LSME-20), it was designed that some or all of exponential parameters are able to be fixed to the input values throughout the execution of the program and also that exponential parameters are arranged according to the order of relative magnitude, namely $\alpha_i > \alpha_{i+1}$ ($i=1,2,,,n-1$) and that during the iteration the order may not be changed.

At the beginning, the program (LSME-20) defines some procedures (or subroutines). NRMEQ is to build up normal equation NQ from exponential Table ET, SLEQ to solve a set of simultaneous equations, UNIT to construct unit matrix which is necessary for estimating the standard errors of parameters and CTHLF to re-evaluate exponential parameters by reducing the correction factor PD to one half.

```
begin comment LSME-20;
  integer EI, GR, GR1, I, J, K, L, LR, M, N,
          NF, NL, NP, NP1, NP2, NP3, NP4,
          NR, NR1, NR2, TP, WI;
  real PR, SP, SN, SS, SM, X, Y, Z;
  array DT [1: 30, 1: 6], ED, T, WT [1: 30], PD,
        PT, PS [1: 5], ET [1: 30, 1: 11], NQ [1: 5,
        1: 16];
  Format F1 [('#')↑10, 'LSME-20-FIT#OF#DATA#',
             10, '#TO', 2, '#(',2,')#EXPONENTIAL#
             EQUATION## (GR=',2,'#WT=',2,')'],
           F2 [('#')↑10, (3, ('#') ↑5) ↑2, 1.6_{10}-2],
           F3 [('#')↑38, (('#') ↑4,-1.4_{10}-2) ↑5],
           F4 [('#')↑15. 'PARAMETER# (',2,')',
              ('##',-5.6,'(',-5.6,')') ↑2],
           F5 [('#')↑15, 5.4, (('#') ↑10, -5.6) ↑3],
           F6 [('#')↑78, 'SS#',-5.6],
           F7 [('#')↑5, 5.4, ('#') ↑5,-5.6];
  procedure SLEQ (A, N, M);
    value N, M; integer N, M; array A;
    begin
      integer I, J, K; real Y;
    for I:=1 step 1 until N do
      begin
        Y:=A[I, I];
      for J:=I step 1 until M do
        A[I, J]:=A[I, J]/Y;
      for K:=1 step 1 until N do
        begin
          if K=I then go to SKIP;
          Y:=A[K, I];
        for J:=I step 1 until M do
          A[K, J]:=A[K, J]-Y*A[I, J];
SKIP:
        end;
      end;
    end;
  procedure NRMEQ (A, B, L, M, N);
    value L, M, N; integer L, M, N;
    array A, B;
    begin
      integer I, J, K;
      real Y;
    for I:=1 step 1 until L do
    for J:=I step 1 until M do
      begin
        Y:=0.0;
      for K:=1 step 1 until N do
        Y:=Y+A[K, I]*A[K, J];
        B[I, J]:=Y;
        end;
      for I:=2 step 1 until L do
      for J:=1 step 1 until I-1 do
        B[I, J]:=B[J, I];
      end;
  procedure CTHLF (A, B, N);
    value N; integer N;
    array A, B;
    begin
      integer I;
    for I:=1 step 1 until N do
      begin
        B[I]:=B[I]*0.5;
          A[I]:=A[I]-B[I];
        end;
      end;
    procedure UNIT (A, N, M);
      value N, M; integer N, M;
      array A;
      begin
        integer I, J;
      for I:=1 step 1 until N do
        begin
        for J:=1 step 1 until N do
          A[I, M+J]:=0.0;
          A[I, M+I]:=1.0;
        end;
      end;
START:
    Read 10 (EI, NP, NF, N, TP, GR, WI, SP, PR,
        LR);
    Print (F1, EI, NP, NF, GR, WI);
    Feed (1);
    Read array (PT[1: NP]);
    Read array (T[1: N]);
    Read array (ED[1: N]);
    if WI=0 then
      for I:=1 step 1 until N do
        WT[I]:=1.0
    else
      begin
        if WI=1 then
          for I:=1 step 1 until N do
            WT[I]:=1.0/sqrt(ED[I])
          else Read array (WT[1: N]);
      end;
    EI:=0;
    NL:=2;
    NR:=NP-NF;
    NR1:=NR+1;
    NR2:=NR1+NR;
    NP1:=NP+1;
    NP2:=NP1+NP;
    NP3:=NP-GR;
    NP4:=NP2+NP;
    GR1:=GR+1;
    SS:=9.9_{10}45;
    for I:=1 step 1 until NR do
      PD [I]:=0.0;
    for L:=1 step 1 until LR do
      begin
        M:=0;
      for I:=1 step 1 until NR do
        PT[I]:=PT[I]+PD[I];
TEST:
        X:=PT[1];
      for I:=2 step 1 until NR do
        begin
          if PT[I]≧X ∨ PT[I]≦0.0 then
            begin
              M:=M+1;
              if M>20 then
              begin
                EI:=1;
                go to LOOP;
              end;
            CTHLF (PT, PD, NR);
            go to TEST;
          end;
        X:=PT[I];
      end;
LOOP:
  SM:=SN:=0.0;
  for I:=1 step 1 until N do
    begin
      X:=WT[I];
      Y:=T[I];
      ET[I, NP1]:=ED[I]*X;
    for J:=1 step 1 until NP do
      begin
        Z:=Y*PT[J];
        if Z>100.0 then Z:=0.0
                   else Z:=exp(-Z)*X;
        if TP=3 then ET[I, J]:=Z-X
                else ET[I, J]:=Z;
        ET[I, NP1+J]:=Y*Z;
      end;
    end;
  NRMEQ (ET, NQ, NP, NP2, N);
  UNIT (NQ, NP, NP2);
    if TP=2 then
      begin
        for I:=1 step 1 until NP do
        for J:=1 step 1 until NP4 do
          NQ[I,J]:=NQ[I,J]-NQ[NP,J];
        for J:=1 step 1 until NP do
          NQ[NP,J]:=1.0;
        end;
    else if TP=3 ∧ GR>0 then
      begin
        for I:=1 step 1 until GR do
        begin
          for J:=1 step 1 until GR do
            DT[I,J]:=PT[NP3+J]↑I;
          for J:=1 step 1 until NP3 do
            DT[I, GR+J]:=-PT[J]↑I;
        end;
    SLEQ (DT, GR, NP);
    for I:=1 step 1 until NP3 do
    for J:=1 step 1 until NP4 do
    for K:=1 step 1 until GR do
      NQ[I, J]:=NQ[I, J]+NQ[NP3+K, J]*DT[K,
                GR+I];
    for I:=1 step 1 until GR do
      begin
        K:=NP3+I;
        for J:=1 step 1 until NP do
        begin
          NQ[K, J]:=PT[J]↑I;
          NQ[K, NP1+J]:=(-1.0)*float(I)*PT[J]
                        ↑(I-1);
          NQ[K, NP2+J]:=0.0;
        end;
      NQ[K, NP1]:=0.0;
      end;
    end;
    SLEQ (NQ, NP, NP4);
    for I:=1 step 1 until N do
    begin
      X:=0.0;
      for J:=1 step 1 until NP do
      X:=X+ET[I, J]*NQ[J, NP1];
      Y:=ET[I, NP1]-X;
      ET[I, NP1]:=X;
      DT[I, NR1]:=Y;
      SN:=SN+Y↑2;
      SM:=SM+(Y/WT[I])↑2;
    end;
    if NR=0 then go to EXIT;
    if SN>SS then
    begin
      M:=M+1;
      if M>20 then EI:=1;
      CTHLF (PT, PD, NR);
      go to LOOP;
    end;
    if abs (SN-SS)<SS* 1.0_{10}-6∨EI=1∨L=LR
      then go to EXIT;
    SS:=SN;
    Print (F2, L, M, SS);
    for I:=1 step 1 until NP do
      Print (F3, PT[I]);
      Print out;
    for I:=1 step 1 until NP do
      Print (F3, NQ[I, NP1]);
      Print out;
    NL:=NL+3;
    for I:=1 step 1 until N do
    for J:=1 step 1 until NR do
    begin
      X:=0.0;
      for K:=1 step 1 until NP do
        X:=X+NQ[K, NP1+J]*ET[I, K];
      DT[I, J]:=(X-ET[I, NP1+J])*NQ[J, NP1];
    end;
    NRMEQ (DT, NQ, NR, NR1, N);
    UNIT (NQ, NR, NR1);
    SLEQ (NQ, NR, NR2);
    EI:=1;
    for I:=1 step 1 until NR do
    begin
      PD[I]:=NQ[I, NR1];
      PS[I]:=NQ[I, NR1+I];
      if abs (PD[I])>PT[I]*0.0001 then EI:=0;
    end;
  end;
EXIT:
    if N≦NR+NP then X:=1.0 else X:=SM/float
      (N-NR-NP);
    Feed (1);
    SN:=0.0;
    for I:=1 step 1 until NP do
    begin
      SN:=SN+NQ[I, NP1];
      Z:=sqrt(NQ[I, NP2+I]*X);
      if I≦NR then PS[I]:=sqrt(PS[I]*X) else
        PS[I]:=0.0;
      Print (F4, I, PT[I], PS[I], NQ[I, NP1], Z);
    end;
    Feed (1);
    Print string[('#')↑19, 'T', ('#')↑19, 'OBS.',('#')
              ↑19, 'CALC.', ('#')↑19, 'DIF.'];
```

```
Feed (1);
SS:=0.0;
for I:=1 step 1 until N do
begin
  X:=1.0/WT[I];
  Y:=DT[I, NR1]*X;
  Print (F5, T[I], ED[I], ET[I, NP1]*X,Y);
  SS:=SS+Y↑2;
end;
Feed (1);
Print (F6, SS);
Feed (5);
Print string [('#')↑5, 'THEORETICAL#
            VALUES'];
Feed (1);
Print string [('#')↑10, 'T', ('#')↑15, 'C'];
Feed (1);
 for X:=0.0 step SP until PR do
 begin
   Y:=0.0;
   for I:=1 step 1 until NP do
   begin
     Z:=X*PT[I];
     if Z>100.0 then Z:=0.0 else Z:=exp(-Z);
     Y:=Y+NQ[I, NP1] *Z;
   end;
   if TP=3 then Y:=Y-SN;
   Print (F7, X, Y);
 end;
 NL:=NL+NP+N+16+entire (PR/SP);
 Feed (60-NL+60*(NL÷60));
 go to START;
end;
```

Fig. 1. Computer Program in ALGOL for Multiexponential Equation

TABLE I. A Description and Explanation of the Important Symbols in LSME-20

Type	Symbol[a]	Comments
Integer	EI	experiment number, exit index
	GR	grade of the equation
	L	interation count
	LR	limit to number of iterations
	M	correction count
	N	number of experimental data
	NF	number of fixed parameters
	NL	number of printed lines
	NP	number of exponential terms
	NP1	NP+1
	NP2	NP+NP1
	NP4	NP+NP2
	NR	number of parameters to be adjusted
	TP	type of the equation
	WI	weight index
Real	PR	period of calculated value print out
	SP	time interval of calculated value
	SS	weighted sum of squared differences
	SN	current value of SS
Array	DT [N×NPI]	differential table
	ED [N]	experimental data
	ET [N×NP2]	exponential table
	NQ [NP×NP4]	normal equation
	PD [NP]	correction to parameter
	PT [NP]	exponential parameter
	PS [NP]	standard error of parameter
	T [N]	sample time of experimental data
	WT [N]	weighting factor

a) The dimensions of the arrays are given within brackets.

Then LSME-20 reads in such informations as data identification number EI, number of exponential terms NP, number of parameters to be fixed NF *etc.*, followed by estimated values for exponential parameters and experimental data.

After initializing some of variables, iteration starts. Evaluation of current parameters is followed by a process to ascertain that the order of the relative magnitude of parameters is not changed. If it is, CTHLF is called. Exponential terms are evaluated and exponential table ET is constructed, then NRMEQ is called to make normal equation NQ, which corresponds to Eq. 8 and 14 for type 1, to Eq. 15 and 16 for type 2 and to Eq. 17 and 18 for type 3.

SLEQ being called for, normal equation is solved to obtain A_j and $\partial A_j/\partial \alpha_l$. Using A_j, theoretical values are calculated as well as sum of squared differences SN. If termination condition is fulfilled, then computation step is transferred to EXIT, otherwise normal equation Eq. 12 is constructed and solved for correction factor PD and the iteration proceeds to another cycle.

On getting out of iteration, standard errors of individual parameters are calculated and printed out with other necessary informations.

Whole program of LSME-20 is shown in Fig. 1. Although it is written in ALGOL, conversion into another language, FORTRAN for example, is not difficult. A description and explanation of the more important symbols in LSME-20 is given in Table I.

Result and Discussion

Several sets of pharmacokinetic data obtained from the literature were fitted by this program. The constants are presented in Table II. For the example of type 1, coefficients and exponential parameters for sulfisoxazole plasma level after *i.v.* administration of the drug calculated by Kaplan, *et al.*[5] were verified. Using averages of the published data (seven subjects), A_1, A_2, α_1 and α_2 calculated by LSME-20 were 70.61 μg/ml, 157.81 μg/ml, 1.023 hr^{-1}, and 0.130 hr^{-1}, and those reported were 108.43 μg/ml, 152.13 μg/ml, 1.393 hr^{-1} and 0.12 hr^{-1}, respectively (average of seven values).

Table II. Computer Calculated Constants from Literature Data

Ref.	NP	Type	Grade	i	A_i[a]	α_i hr^{-1}
(5)	2	1	—	1	70.6135± 2.9401	1.0226±0.1474
				2	157.8052± 1.4353	0.1300±0.0065
(6)	2	2	—	1	−11.6099± 0.1474	3.9549±0.3339
				2	11.6099± 0.1474	0.1541±0.0184
(7) S-F	1	3	0	1	−386.6921± 1.8819	0.1144±0.0024
S-A	2 (free)	3	0	1	20.8414± 2.5232	0.5857±0.4407
				2	−264.2627± 3.7425	0.0524±0.0035
S-A	2 (fix)	3	0	1	194.0952±13.0753	0.1144 (fixed)
				2	−431.8884±14.5543	0.0696±0.0077
I-F	1	3	0	1	−462.3187± 3.9608	0.1071±0.0039
I-A	2 (free)	3	0	1	35.2883± 0.6843	0.3979±0.0471
				2	−226.7945± 0.9743	0.0541±0.0013
I-A	2 (fix)	3	0	1	606.2616±20.3476	0.1071 (fixed)
				2	−791.5451±21.2973	0.0858±0.0073

a) Dimensions of the values depend on the source data.

Blood concentrations of subject Bs from SETD given in 2.0 g dose reported by Swintosky, *et al.*[6] were used as type 2 data. On fitting the same data to Eq. 20, which is essentially two-exponential equation, by grid search method, Lowenthal, *et al.*[3g] omitted the concentration at 20 minutes, because there was an apparent lag time in the initial absorption of the drug before absorption becomes an exponential process. The authors of the present report subtracted 15 minutes from each of time data, instead, to take the lag time into account.

$$C=\frac{K_aA}{V_d(K_a-K_d)}(e^{-K_dt}-e^{-K_at}) \qquad \text{Eq. 20}$$

Urinary excretion of sulfisoxazole and acetylsulfisoxazole reported by Nelson and O' Reilly[7]

5) S.A. Kaplan, R.E. Weinfeld, C.W. Abruzzo, and M. Lewis, *J. Pharm. Sci.*, **61**, 773 (1972).
6) J.V. Swintosky, E.L. Foltz, A. Bondi Jr., and M.J. Robinson, *J. Am. Pharm. Assoc., Sci. Ed.*, 47, 136 (1958).
7) E. Nelson and O'Reilly, *J. Pharmacol. Exptl. Therap.*, **129**, 368 (1960).

(subjects S and I) were used as type 3 data, after subtracting amounts of drug excretion in the first 4 hours following drug ingestion from the comulative amounts found at subsequent times. This procedure, a shift in zero time, corrects the data for postabsorptive and post-equilibrative phases of drug excretion and metabolism, according to the authors. Elimination rate constant of sulfisoxazole calculated by Nelson, *et al.* was verified. There was essentially no difference between the elimination rate constant of acetylsulfisoxazole calculated by Nelson, *et al.* and one of the two exponential parameters computed by the program of the present paper if the other parameter was fixed to the value of the elimination rate constant of sulfisoxazole just obtained above. Free of such a restriction, acetylsulfisoxazole data fitted better to a two-exponential equation, whose parameters were a little different as shown in Table II.